ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ОРЕНБУРГСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

Б1.В.ДВ.06.02 Пакеты прикладных программ

Направление подготовки (специальность) 38.03.02 Менеджмент

Профиль образовательной программы Управленческий и финансовый учет

Форма обучения очная

СОДЕРЖАНИЕ

1. КОНСПЕКТ ЛЕКЦИЙ

1. 1 Лекция №1 (2 часа).

Тема: «Введение в предмет. Понятие «Пакеты прикладных программ»»

1.1.1 Вопросы лекции:

1. Краткая характеристика дисциплины
2. История создания дисциплины

1.1.2 Краткое содержание вопросов:

1. Краткая характеристика дисциплины.

Целями освоения дисциплины «Пакеты прикладных программ» являются:

- сформировать представление о принципах моделирования и методах решения задач управления с помощью автоматизированных информационных технологий;

- ознакомить студентов с методами решения задач, связанных с прогнозированием и планированием производства (в том числе сельскохозяйственного), размещения денежных вкладов и пр. с целью анализа и обоснования принятия верных управленческих решений.

Задачи дисциплины являются

- изучение основных принципов, используемых в разработке интегрированных программных продуктов;
- изучение структуры, состава и назначения компонентов интегрированного ПО, а также средств организации взаимодействия между компонентами и инструментальных средств расширения функциональности;
- формирование навыков работы со средствами автоматизации решения прикладных задач;
- формирование навыков использования встроенных средств разработки.

Компетенции, формируемые в результате освоения дисциплины:

- способностью использовать основные методы финансового менеджмента для стоимостной оценки активов, управления оборотным капиталом, принятия решений по финансированию, формированию дивидендной политики и структуре капитала (ПК-11);
- способностью проводить оценку инвестиционных проектов при различных условиях инвестирования и финансирования (ПК-43);
- способностью разрабатывать бизнес-планы создания и развития новых организаций (направлений деятельности, продуктов) (ПК-49).

В результате освоения дисциплины обучающийся должен:

- *знать*: основные приемы применения пакетов прикладных программ в экономических расчетах; методические подходы к формализации прикладных экономических задач финансовой сферы деятельности, вопросы использования результатов решения задач при принятии эффективных управленческих решений;
- *уметь*: работать с конкретными программными продуктами, используемыми в финансово-экономической сфере; использовать их для решения экономико-математических моделей;
- *владеть*: информацией о классификации экономико-математических методов относительно их применения к решению экономических задач и возможности их применения в управлении финансово-экономическими учреждениями; специальной терминологией, навыками самостоятельного овладения новыми знаниями.

Основные понятия и определения

Информационная система (ИС) — организационно упорядоченная совокупность документов (массивов документов) и информационных технологий, в том числе с использованием средств вычислительной техники и связи, реализующих информационные процессы.

Информационные системы предназначены для хранения, обработки, поиска, распространения, передачи и представления информации.

Автоматизированная (информационная) система (АС) — совокупность программных и аппаратных средств, предназначенных для хранения и/или управления данными и информацией и производства вычислений и управляемая человеком-оператором (в этом главное отличие автоматизированной системы от автоматической).

Многоуровневое представление ИС — модель представления информационной системы в виде совокупности взаимосвязанных уровней, разделенных по функциональному назначению (рис. 1).

Рисунок 1 - Многоуровневое представление информационных систем.

Аппаратное обеспечение ИС — комплекс электронных, электрических и механических устройств, входящих в состав информационной системы или сети.

Программное обеспечение (ПО) — совокупность программ и данных, предназначенных для решения определенного круга задач и хранящиеся на машинных носителях.

Программа — последовательность формализованных инструкций, представляющих алгоритм решения некоторой задачи и предназначенная для исполнения устройством управления вычислительной машины. Инструкции программы записываются при помощи машинного кода или специальных языков программирования. В зависимости от контекста термин «программа» может относится к исходным текстам, при помощи которых записывается алгоритм, или к исполняемому машинному коду.

Программист — специалист, занимающийся разработкой и проверкой программ. Различают системных и прикладных программистов.

Пользователь — человек, принимающий участие в управлении объектами и системами некоторой предметной области и являющийся составным элементом автоматизированной системы.

Прикладное программное обеспечение — программное обеспечение, ориентированное на конечного пользователя и предназначенное для решения пользовательских задач. Прикладное ПО состоит из:

отдельных прикладных программ и пакетов прикладных программ, предназначенных для решения различных задач пользователей;

автоматизированных систем, созданных на основе этих пакетов.

Пакет прикладных программ — комплект программ, предназначенных для решения задач из определенной проблемной области. Обычно применение пакета прикладных

программ предполагает наличие специальной документации: лицензионного свидетельства, паспорта, инструкции пользователя и т.п.

Понятие пакета прикладных программ

Пакет прикладных программ (ППП) – это комплекс взаимосвязанных программ для решения определенного класса задач из конкретной предметной области. На текущем этапе развития информационных технологий именно ППП являются наиболее востребованным видом прикладного ПО. Это связано с упомянутыми ранее особенностями ППП. Рассмотрим их подробней:

Ориентация на решение класса задач. Одной из главных особенностей является ориентация ППП не на отдельную задачу, а на некоторый класс задач, в том числе и специфичных, из определенной предметной области. Так например, офисные пакеты ориентированы на офисную деятельность, одна из задач которой — подготовка документов (в общем случае включающих не только текстовую информацию, но и таблицы, диаграммы, изображения). Следовательно, офисный пакет должен реализовывать функции обработки текста, представлять средства обработки табличной информации, средства построения диаграмм разного вида и первичные средства редактирования растровой и векторной графики.

Наличие языковых средств. Другой особенностью ППП является наличие в его составе специализированных языковых средств, позволяющих расширить число задач, решаемых пакетом или адаптировать пакет под конкретные нужды. Пакет может представлять поддержку нескольких входных языков, поддерживающих различные парадигмы. Поддерживаемые языки могут быть использованы для формализации исходной задачи, описания алгоритма решения и начальных данных, организации доступа к внешним источникам данных, разработки программных модулей, описания модели предметной области, управления процессом решения в диалоговом режиме и других целей. Примерами входных языков ППП являются VBA в пакете MS Office, AutoLISP/VisualLISP в Autodesk AutoCAD, StarBasic в OpenOffice.org

Единообразие работы с компонентами пакета. Еще одна особенность ППП состоит в наличии специальных системных средств, обеспечивавших унифицированную работу с компонентами. К их числу относятся специализированные банки данных, средства информационного обеспечения, средства взаимодействия пакета с операционной системой, типовой пользовательский интерфейс и т.п.

Современные прикладные пакеты — это сложные программные решения, реализующие множество функций. В ходе проектирования и разработки ППП эти функции группируются по назначению и объединяются в структурные компоненты. Можно выделить по меньшей мере три таких компонента ППП: входной язык, предметное обеспечение и системное обеспечение.

Со времени появления первых компьютеров появилось множество прикладных разработок, но, несмотря на разнообразие, их обобщенную внутреннюю структуру можно представить в виде трех взаимосвязанных элементов (рис. 2):

- входной язык (макроязык, язык управления) — представляет средство общения пользователя с пакетом;
- предметное обеспечение (функциональное наполнение) — реализует особенности конкретной предметной области;
- системное обеспечение (системное наполнение) — представляет низкоуровневые средства, например, доступ к функциям операционной системы.

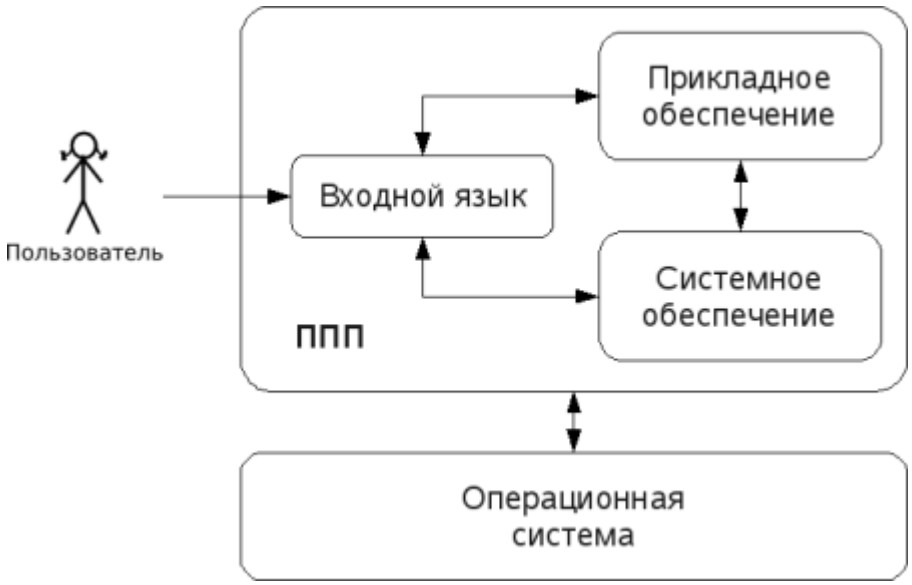


Рисунок 2 - Структура ППП

Входной язык — основной инструмент при работе пользователя с пакетом прикладных программ. В качестве входного языка могут использоваться как универсальные (Pascal, Basic и т.п.), так и специализированные, проблемно-ориентированные языки программирования (Cobol — для бизнес-приложений, Lisp — списочные структуры данных, Fortran и MathLAB — математические задачи и т.п.).

Развитый пакет может обладать несколькими входными языками, предназначенными для выполнения различных функций в рамках решаемого класса задач. Так, например в пакете OpenOffice.org поддерживаются языки StarBasic, Python, JavaScript и Java. StarBasic является основным входным языком, предназначенным для автоматизации работы с пакетом, для этого языка имеется интегрированная среда разработки и встроенный отладчик. Скрипты на языках Python и JavaScript загружаются и исполняются из внешних файлов. На Java (через SDK и функции API OpenOffice) можно создавать модули расширения и полнофункциональные приложения-компоненты.

Входные языки отражают объем и качество предоставляемых пакетом возможностей, а также удобство их использования. Таким образом, именно входной язык является основным показателем возможностей ППП. Однако стоит отметить, что в современных пакетах обращение пользователя к языковым средствам обычно происходит косвенно, через графический интерфейс.

Предметное обеспечение отражает особенности решаемого класса задач из конкретной предметной области и включает:

- программные модули, реализующие алгоритмы (или их отдельные фрагменты) прикладных задач;
- средства сборки программ из отдельных модулей.

Наиболее распространено в настоящее время оформление программных модулей в виде библиотек, подключаемых статически или динамически. В зависимости от использованного разработчиками подхода к проектированию и реализации ППП такие библиотеки содержат встроенные классы и описания их интерфейсов (при использовании объектно-ориентированного программирования). При использовании парадигмы структурного программирования в библиотечных модулях содержатся процедуры и функции, предназначенные для решения некоторых самостоятельных задач. В обоих случаях библиотеки связаны с другими модулями пакета лишь входной и выходной информацией.

Системное обеспечение представляет собой совокупность низкоуровневых средств (программы, файлы, таблицы и т.д.), обеспечивающих определенную дисциплину работы пользователя при решении прикладных задач и формирующих окружение пакета. К системному обеспечению ППП относят следующие компоненты:

-монитор — программа, управляющая взаимодействием всех компонентов ППП;

-транслятор(ы) с входных языков — для ППП характерно использование интерпретируемых языков;

-средства доступа к данным — драйверы баз данных и/или компоненты, представляющие доступ через унифицированные интерфейсы (ODBC, JDBC, ADO, BDE и т.п.);

-информационно-справочный модуль — предоставляет функции поддержки, среди которых информационные сообщения, встроенная справочная системы и т.п.

-различные служебные программы, выполняющие низкоуровневые операции (автосохранение, синхронизация совместно используемых файлов и т.д.)

Приведенная трехкомпонентная логическая структура ППП достаточна условна, она зависит от использованных подходов к проектированию ПО, используемым технологиям программирования, предметной области и других факторов, вплоть до индивидуальных предпочтений разработчика. Так, в конкретном ППП может отсутствовать четкое разделение программ на предметное и системное обеспечение. Например, программа планирования вычислений, относящаяся к прикладному обеспечению, может одновременно выполнять и ряд служебных функций (информационное обеспечение, связь с операционной системой и т.п.). С другой стороны, распределенные приложения добавляют свою специфику в структуру ППП.

Кроме того, одни и те же программы в одном пакете могут относиться к предметному обеспечению, а в другом — к системному. Так, программы построения диаграмм в рамках специализированного пакета машинной графики естественно отнести к предметному обеспечению. Однако те же программы следует считать вспомогательными и относящимися к системному обеспечению, например, в пакете решения вычислительных задач.

2. История создания дисциплины.

Этапы развития ППП

Первые пакеты прикладных программ представляли собой простые тематические подборки программ для решения отдельных задач в той или иной предметной области, обращение к ним выполнялось с помощью средств оболочки ОС или из других программ. Современный пакет является сложной программной системой, включающей специализированные системные и языковые средства. В относительно короткой истории развития вычислительных ППП можно выделить *4 основных поколения* (класса) пакетов. Каждый из этих классов характеризуется определенными особенностями входящих состав ППП компонентов — входных языков, предметного и системного обеспечения.

Первое поколение

В качестве входных языков ППП первого поколения использовались универсальные языки программирования (Фортран, Алгол-60 и т.п.) или языки управления заданиями соответствующих операционных систем. Проблемная ориентация входных языков достигалась за счет соответствующей мнемоники в идентификаторах. Составление заданий на таком языке практически не отличалось от написания программ на алгоритмическом языке. Предметное обеспечение первых ППП, как правило, было организовано в форме библиотек программ, т.е. в виде наборов (пакетов) независимых программ на некотором базовом языке программирования (отсюда впервые возник и сам термин «пакет»). Такие ППП иногда называют *пакетами библиотечного типа*, или *пакетами простой структуры*.

В качестве системного обеспечения пакетов первого поколения обычно использовались штатные компоненты программного обеспечения ЭВМ: компиляторы с алгоритмических языков, редакторы текстов, средства организации библиотек программ, архивные системы и т.д. Эти пакеты не требовали сколь-нибудь развитой системной поддержки, и для их функционирования вполне хватало указанных системных средств общего назначения. В большинстве случаев разработчиками таких пакетов были прикладные

программисты, которые пытались приспособить универсальные языки программирования к своим нуждам.

Второе поколение

Разработка ППП второго поколения осуществлялась уже с участием системных программистов. Это привело к появлению специализированных входных языков на базе универсальных языков программирования. Проблемная ориентация таких языков достигалась не только за счет использования определенной мнемоники, но также применением соответствующих языковых конструкций, которые упрощали формулировку задачи и делали ее более наглядной. Транслятор с такого языка представлял собой препроцессор (чаще всего макропроцессор) к транслятору соответствующего алгоритмического языка. В качестве модулей в пакетах этого класса стали использоваться не только программные единицы (т.е. законченные программы на том или ином языке программирования), но и такие объекты, как последовательность операторов языка программирования, совокупность данных, схема счета и др.

Существенные изменения претерпели также принципы организации системного обеспечения ППП. В достаточно развитых пакетах второго поколения уже можно выделить элементы системного обеспечения, характерные для современных пакетов: монитор, трансляторы с входных языков, специализированные банки данных, средства описания модели предметной области и планирования вычислений и др.

Третье поколение

Третий этап развития ППП характеризуется появлением самостоятельных входных языков, ориентированных на пользователей-непрограммистов. Особое внимание в таких ППП уделяется системным компонентам обеспечивающим простоту и удобство. Это достигается главным образом за счет специализации входных языков и включения в состав пакета средств автоматизированного планирования вычислений.

Интероперабельность ППП

Расширение сферы применения вычислительной техники привело к появлению разнообразных цифровых устройств (от настольных систем и нетбуков, до смартфонов и бытовой техники) на различных аппаратно-программных платформах. Эта ситуация, с одной стороны, обострила вопрос, возникший еще у разработчиков вычислительных систем первого-второго поколений: как обеспечить работу программ при смене платформы? А с другой — создала проблему для конечных пользователей: как использовать необходимые им ППП, созданные для другой системы и при этом не менять платформу? То есть как, например, заставить работать приложения для андроид под управлением iOS или как запустить MS Office под Linux'ом.

Решением проблемы обеспечения *интероперабельности* (переносимости) ПО является концепция *открытых систем*. Такая система разрабатывается с использованием *открытых стандартов и спецификаций*. Эта концепция все чаще используется разработчиками прикладного программного обеспечения и существенно облегчает жизнь пользователям. Примерами такого ПО являются офисный пакет LibreOffice, веб-браузер Firefox и всевозможные веб-сервисы.

Четвертое поколение

Четвертый этап характеризуется созданием ППП, эксплуатируемых в интерактивном режиме работы. Основным преимуществом диалогового взаимодействия с ЭВМ является возможность активной обратной связи с пользователем в процессе постановки задачи, ее решения и анализа полученных результатов. Появление и интенсивное развитие различных форм диалогового общения обусловлено прежде всего прогрессом в области технических средств (графическая подсистема ЭВМ и средства мультимедиа, сетевые средства).

Развитие аппаратного обеспечения повлекло за собой создание разнообразных программных средств поддержки диалогового режима работы (диалоговые операционные системы, диалоговые пакеты программ различного назначения и т. д.). Прикладная система состоит из *диалогового монитора* — набора универсальных программ, обеспечивающих

ведение диалога и обмен данными, и *базы знаний* о предметной области. Информация о структуре, целях и форма диалога задает *сценарий*, в соответствии с который монитор управляет ходом диалога. Носителями процедурных знаний о предметной области являются прикладные модули, реализующие функции собственно системы.

Таким образом, создание прикладной системы сводится к настройке диалогового монитора на конкретный диалог, путем заполнения базы знаний. При этом программировать в традиционном смысле этого слова приходится лишь прикладные модули, знания о диалоге вводятся в систему с помощью набора соответствующих средств — редактора сценариев. Логично требовать, чтобы редактор сценариев также представлял собой диалоговую программу, отвечавшую рассмотренным выше требованиям. Благодаря готовому универсальному монитору программист может сосредоточиться на решении чисто прикладных задач, выделение же знаний о диалоге в сценарий обеспечивает в значительной степени необходимая гибкость программного продукта.

Большое внимание в настоящее время уделяется проблеме создания «*интеллектуальных ППП*». Такой пакет позволяет конечному пользователю лишь сформулировать свою задачу в содержательных терминах, не указывая алгоритма ее решения. Синтез решения и сборка целевой программы производятся автоматически. При этом детали вычислений скрыты от пользователя, и компьютер становится интеллектуальным партнером человека, способным понимать его задачи. Предметное обеспечение подобного ППП представляет собой некоторую базу знаний, содержащую как процедурные, так и описательные знания. Такой способ решения иногда называют концептуальным программированием, характерными особенностями которого является программирование в терминах предметной области использование ЭВМ уже на этапе постановки задач, автоматический синтез программ решения задачи, накопление знаний о решаемых задачах в базе знаний.

Примеры современных прикладных пакетов

Рассматриваются несколько прикладных пакетов от разных разработчиков. Приводятся краткие описания возможностей структурных компонентов этих ППП.

Для иллюстрации ранее изученных материалов приведем несколько примеров современных пакетов прикладных программ из различных предметных областей. Учитывая, что постоянно появляются новые версии программных продуктов, здесь будут рассматриваться не возможности конкретных версий, а лишь основные структурные компоненты, входящие в состав того или иного пакета.

Autodesk AutoCAD

Основное назначение системы автоматизированного проектирования Autodesk AutoCAD — создание чертежей и проектной документации. Современные версии этого пакета представляют существенно большие возможности, среди которых построение трехмерных твердотельных моделей, инженерно-технические расчеты и многое-многое другое.

Первые версии системы AutoCAD, разрабатываемой американской фирмой Autodesk, появились еще в начале 80-х годов двадцатого века, и сразу же привлекли к себе внимание своим оригинальным оформлением и удобством для пользователя. Постоянное развитие системы, учет замечаний, интеграция с новыми продуктами других ведущих фирм сделали AutoCAD мировым лидером на рынке программного обеспечения для автоматизированного проектирования.

В основе языковых средств ППП AutoCAD — технология Visual LISP, базирующаяся на языке AutoLISP (подмножество языка LISP) и используемая для создания приложений и управления в AutoCAD. Visual LISP представляет полное окружение, включающее:

Интегрированную среду разработки, облегчающую написание, отладку и сопровождение приложений на AutoLISP

Доступ к объектам ActiveX и обработчикам событий

Защиту исходного кода

Доступ к файловым функциям операционной системы

Расширенные функции языка LISP для обработки списочных структур данных.

Для разработчиков совместимых приложений в AutoCAD включена поддержка ObjectARX. Это программное окружение представляет объектно-ориентированный интерфейс для приложений на языках C++, C# и VB.NET и обеспечивает прямой доступ к структурам БД, графической подсистеме и встроенным командам пакета.

Кроме того, в AutoCAD имеется поддержка языка Visual Basic for Applications (VBA), что позволяет использовать этот пакет совместно с другими приложениями, в частности, из семейства Microsoft Office.

К предметному обеспечению пакета в первую очередь относятся функции построения примитивов — различных элементов чертежа. Простые примитивы это такие объекты как точка, отрезок, круг (окружность) и т.д. К сложным примитивам относятся: полилиния, мультилиния, мультитекст (многострочный текст), размер, выноска, допуск, штриховка, вхождение блока или внешней ссылки, атрибут, растровое изображение. Кроме того, есть пространственные примитивы, видовые экраны и пр.. Операции построения *большей части* примитивов могут быть выполнены через пользовательский интерфейс, *все* — через команды языка.

Высокоуровневые средства представлены расширениями и приложениями AutoCAD для конкретных предметных областей. Например в машиностроении используется Autodesk Mechanical Desktop — предназначенный для сложного трехмерного моделирования, в том числе валов и пружин. Для проектирования деталей из листовых материалов предназначена система Copra Sheet Metal Bender Desktop (разработчик — Data-M Software GmbH). Моделирование динамики работы механизмов может выполняться в системе Dynamic Designer (Mechanical Dynamics). В числе известных архитектурных и строительных приложений можно отметить системы АРКО (АПИО-Центр), СПДС GraphiCS (Consistent Software), ArchiCAD. Для проектирования промышленных объектов может использоваться система PLANT-4D (CEA Technology). Это лишь некоторые из областей использования AutoCAD.

Среди системного обеспечения следует отметить основной формат файлов AutoCAD .dwg, который стал стандартом «де факто» для прочих САПР.

К системному же обеспечению обеспечению относятся типовые и специализированные библиотеки деталей и шаблонов, использование которых позволяет существенно ускорить процесс проектирования. Здесь же упомянем требования отраслевых и государственных стандартов, которым должны соответствовать чертежи и спецификации.

Конфигурация и настройки различных режимов AutoCAD устанавливаются через т.н. системные переменные. Изменяя их значения можно задавать пути к файлам, точность вычислений, формат вывода и многое другое.

Adobe Flash

Adobe (ранее Macromedia) Flash — это технология и инструментарий разработки интерактивного содержания с большими функциональными возможностями для цифровых, веб- и мобильных платформ. Она позволяет создавать компактные, масштабируемые анимированные приложения (ролики), которые можно использовать как отдельно, так и встраивая в различное окружение (в частности, в веб-страницы). Эти возможности обеспечиваются следующими компонентами технологии: языком Action Script, векторным форматом .swf и видеоформатом .flv, всевозможными flash-плейерами для просмотра и редакторами для создания.

Рассмотрим интегрированную среду Adobe Flash как основное средство создания flash-приложений. При этом отметим, что языковые и системные средства относятся не только к этому пакету, а к технологии в целом. Если, например, купить фотошоп cs3 в соответствующей конфигурации, то эти средства будут доступны для всех приложений из состава пакета.

ActionScript — объектно-ориентированный язык программирования, который добавляет интерактивность, обработку данных и многое другое в содержимое Flash-приложений. Синтаксис ActionScript основан на спецификации ECMAScript (сюда же относятся языки JavaScript и JScript). Библиотека классов ActionScript, написанная на C++, представляет доступ к графическим примитивам, фильтрам, принтерам, геометрическим функциям и пр..

ActionScript как язык появился с выходом 5 версии Adobe (тогда еще Macromedia) Flash, которая стала первой программируемой на ActionScript средой. Первый релиз языка назывался ActionScript 1.0. Flash 6 (MX). В 2004 году Macromedia представила новую версию ActionScript 2.0 вместе с выходом Flash 7 (MX 2004), в которой было введено строгое определение типов, основанное на классах программирование: наследование, интерфейсы и т. д. Также Macromedia была выпущена модификация языка Flash Lite для программирования под мобильные телефоны. ActionScript 2.0 является не более чем надстройкой над ActionScript 1.0, то есть на этапе компиляции ActionScript 2.0 осуществляет некую проверку и превращает классы, методы ActionScript 2.0 в прежние прототипы и функции ActionScript 1.0.

В 2005 году вышел ActionScript 3.0 в среде программирования Adobe Flex, а позже в Adobe Flash 9.

ActionScript 3.0 (текущая версия на момент подготовки этого материала) представляет, по сравнению с ActionScript 2.0 качественное изменение, он использует новую виртуальную машину AVM 2.0 и дает взамен прежнего формального синтаксиса классов настоящее классовое (class-based) Объектно-ориентированное программирование. ActionScript 3.0 существенно производительней предыдущих версий и по скорости приблизился к таким языкам программирования, как Java и C++.

С помощью ActionScript можно создавать интерактивные мультимедиа-приложения, игры, веб-сайты и многое другое.

ActionScript исполняется виртуальной машиной (ActionScript Virtual Machine), которая является составной частью Flash Player. ActionScript компилируется в байткод, который включается в SWF-файл.

SWF-файлы исполняются Flash Player-ом. Flash Player существует в виде плагина к веб-браузеру, а также как самостоятельное исполняемое приложение. Во втором случае возможно создание исполняемых exe-файлов, когда swf-файл включается во Flash Player.

Для создания и просмотра видеофайлов в формате .flv используются программные кодеки, поддерживающие этот формат.

К прикладному обеспечению в рамках технологии Flash относятся средства создания роликов в форматах .swf, .flv и .exe. Основным инструментом является среда среда Adode Flash (см. Adobe Flash Builder), включающая различные средства для создания и редактирования мультимедийного содержания, в т.ч. видео- и аудиофайлов, интегрированную среду разработки на ActionScript и множество дополнительных функций упрощения процесса создания роликов.

Пакет MatLab

MatLab (сокращение от англ. «Matrix Laboratory») — пакет прикладных программ для решения задач технических вычислений, и язык программирования, используемый в этом пакете. По данным фирмы-разработчика, более 1000000 инженерных и научных работников используют этот пакет, который работает на большинстве современных операционных систем, включая GNU/Linux, Mac OS, Solaris и Microsoft Windows.

Язык MatLab

MATLAB как язык программирования был разработан Кливом Моулером (англ. Cleve Moler) в конце 1970-х годов. Целью разработки служила задача использования программных математических библиотек Linpack и EISPACK без необходимости изучения языка Фортран. Акцент был сделан на матричные алгоритмы.

Программы, написанные на MATLAB, бывают двух типов — функции и скрипты. Функции имеют входные и выходные аргументы, а также собственное рабочее пространство для хранения промежуточных результатов вычислений и переменных. Скрипты же используют общее рабочее пространство. Как скрипты, так и функции не компилируются в машинный код, а сохраняются в виде текстовых файлов. Существует также возможность сохранять так называемые pre-parsed программы — функции и скрипты, приведенные в вид, удобный для машинного исполнения и, как следствие, более быстрые по сравнению с обычными.

Язык MATLAB является высокоуровневым интерпретируемым языком программирования, включающим основанные на матрицах структуры данных, широкий спектр функций, интегрированную среду разработки, объектно-ориентированные возможности и интерфейсы к программам, написанным на других языках программирования. Имеются интерфейсы для получения доступа к внешним данным, клиентам и серверам, общающимся через технологии Component Object Model (COM) или Dynamic Data Exchange (DDE), а также периферийным устройствам, которые взаимодействуют напрямую с MATLAB. Многие из этих возможностей известны под названием MATLAB API.

Встроенная среда разработки позволяет создавать графические интерфейсы пользователя с различными элементами управления, такими как кнопки, поля ввода и другими. С помощью компонента MATLAB Compiler эти графические интерфейсы могут быть преобразованы в самостоятельные приложения.

Для MATLAB имеется возможность создавать специальные наборы инструментов (англ. toolbox), расширяющие его функциональность. Наборы инструментов представляют собой коллекции функций, написанных на языке MATLAB для решения определенного класса задач. С некоторыми, весьма полезными, примерами таких расширений для MatLab можно ознакомиться здесь.

MATLAB предоставляет удобные средства для разработки алгоритмов, включая высокоуровневые с использованием концепций объектно-ориентированного программирования. В нем имеются все необходимые средства интегрированной среды разработки, включая отладчик и профайлер.

MATLAB предоставляет пользователю большое количество (несколько сотен) функций для анализа данных, покрывающие практически все области математики, в частности:

Матрицы и линейная алгебра — алгебра матриц, линейные уравнения, собственные значения и вектора, сингулярности, факторизация матриц и другие.

Многочлены и интерполяция — корни многочленов, операции над многочленами и их дифференцирование, интерполяция и экстраполяция кривых и другие.

Математическая статистика и анализ данных — статистические функции, статистическая регрессия, цифровая фильтрация, быстрое преобразование Фурье и другие.

Обработка данных — набор специальных функций, включая построение графиков, оптимизацию, поиск нулей, численное интегрирование (в квадратурах) и другие.

Дифференциальные уравнения — решение дифференциальных и дифференциально-алгебраических уравнений, дифференциальных уравнений с запаздыванием, уравнений с ограничениями, уравнений в частных производных и другие.

Разреженные матрицы — специальный класс данных пакета MATLAB, использующийся в специализированных приложениях.

В составе пакета имеется большое количество функций для построения графиков, в том числе трехмерных, визуального анализа данных и создания анимированных роликов, функции для создания алгоритмов для микроконтроллеров и других приложений.

1. 2 Лекция №2 (2 часа).

Тема: «Система баз данных»

1.2.1 Вопросы лекции:

1. Определение и назначение баз данных
2. Файловые системы хранения данных
3. Области применения баз данных

1.2.2 Краткое содержание вопросов:

1. Определение и назначение баз данных.

С самого начала развития вычислительной техники образовались два основных направления ее использования.

Первое направление — применение вычислительной техники для выполнения численных расчетов, которые слишком долго или вообще невозможно производить вручную.

Второе направление — это использование средств вычислительной техники в автоматических или автоматизированных информационных системах. В самом широком смысле информационная система представляет собой программный комплекс, функции которого состоят в поддержке надежного хранения информации в памяти компьютера, выполнении специфических для данного приложения преобразований информации и/или вычислений, предоставлении пользователям удобного и легко осваиваемого интерфейса. Обычно объемы информации, с которыми приходится иметь дело таким системам, достаточно велики, а сама информация имеет достаточно сложную структуру. Классическими примерами информационных систем являются банковские системы, системы резервирования авиационных или железнодорожных билетов, мест в гостиницах и т. д.

Второе направление возникло несколько позже первого. Это связано с тем, что на заре вычислительной техники компьютеры обладали ограниченными возможностями. Надежное и долговременное хранение информации возможно только при наличии запоминающих устройств, сохраняющих информацию после выключения электрического питания. Оперативная память этим свойством обычно не обладает. Используемые в ранних ЭВМ два вида устройств внешней памяти, магнитные ленты и барабаны были несовершенными. Емкость магнитных лент была достаточно велика, но по своей физической природе они обеспечивали последовательный доступ к данным. Магнитные барабаны давали возможность произвольного доступа к данным, но были ограниченного размера. Появление соответствующих носителей данных, в первую очередь, жестких дисков, дало толчок к работам по созданию информационных компьютерных систем.

Основу любой информационной системы составляет база данных — это набор данных, которые организованы специальным образом.

В настоящее время действует Закон «О правовой охране программ для электронных вычислительных машин и баз данных» № 3523-1 от 23.09.92. В этом законе дается следующее определение базы данных: «**База данных** — это объективная форма представления и организации совокупности данных (например, статей, расчетов), систематизированных таким образом, чтобы эти данные могли быть найдены и обработаны с помощью ЭВМ».

В основе решения многих задач лежит обработка информации. Для облегчения обработки информации создаются информационные системы (ИС). Автоматизированными называют ИС, в которых применяют технические средства, в частности ЭВМ. Большинство существующих ИС являются автоматизированными, поэтому для краткости просто будем называть их ИС.

В широком понимании под определение ИС подпадает любая система обработки информации. По области применения ИС можно разделить на системы, используемые в производстве, образовании, здравоохранении, науке, военном деле, социальной сфере, торговле и других отраслях. По целевой функции ИС можно условно разделить на следующие основные категории: управляющие, информационно-справочные, поддержки принятия решений.

Заметим, что иногда используется более узкая трактовка понятия ИС как совокупности аппаратно-программных средств, задействованных для решения некоторой прикладной задачи. В организации, например, могут существовать информационные системы, на которых соответственно возложены следующие задачи: учет кадров и материально-технических средств, расчет с поставщиками и заказчиками, бухгалтерский учет и т. п.

Банк данных является разновидностью ИС, в которой реализованы функции централизованного хранения и накопления обрабатываемой информации, организованной в одну или несколько баз данных. Банк данных (БнД) в общем случае состоит из следующих компонентов: базы (нескольких баз) данных, системы управления базами данных, словаря данных, администратора, вычислительной системы и обслуживающего персонала. Вкратце рассмотрим названные компоненты и некоторые связанные с ними важные понятия.

База данных (БД) представляет собой совокупность специальным образом организованных данных, хранимых в памяти вычислительной системы и отображающих состояние объектов и их взаимосвязей в рассматриваемой предметной области.

Логическую структуру хранимых в базе данных называют моделью представления данных. К основным моделям представления данных (моделям данных) относятся следующие: иерархическая, сетевая, реляционная, постреляционная, многомерная и объектно-ориентированная (см. раздел 2).

Система управления базами данных (СУБД) - это комплекс языковых и программных средств, предназначенный для создания, ведения и совместного использования БД многими пользователями. Обычно СУБД различают по используемой модели данных. Так, СУБД, основанные на использовании реляционной модели данных, называют реляционными СУБД.

Одними из первых СУБД являются следующие системы: IMS (IBM, 1968 г.), IDMS (Cullinet, 1971 г.), ADABAS (Software AG, 1969 г.) и ИНЭС (ВНИИСИ АН СССР, 1976 г.). Количество современных систем управления базами данных исчисляется тысячами.

Приложение представляет собой программу или комплекс программ, обеспечивающих автоматизацию обработки информации для прикладной задачи. Нами рассматриваются приложения, использующие БД. Приложения могут создаваться в среде или вне среды СУБД - с помощью системы программирования, использующей средства доступа к БД, к примеру, Delphi или C++ Builder. Приложения, разработанные в среде СУБД, часто называют приложениями СУБД, а приложения, разработанные вне СУБД, - внешними приложениями.

Для работы с базой данных зачастую достаточно средств СУБД и не нужно использовать приложения, создание которых требует программирования. Приложения разрабатывают главным образом в случаях, когда требуется обеспечить удобство работы с БД неквалифицированным пользователям или интерфейс СУБД не устраивает пользователей.

Словарь данных (СД) представляет собой подсистему БнД, предназначенную для централизованного хранения информации о структурах данных, взаимосвязях файлов БД друг с другом, типах данных и форматах их представления, принадлежности данных пользователям, кодах защиты и разграничения доступа и т.п. Функционально СД присутствует во всех БнД, но не всегда выполняющий эти функции компонент имеет именно такое название. Чаще всего функции СД выполняются СУБД и вызываются из основного меню системы или реализуются с помощью ее утилит.

Администратор базы данных (АБД) есть лицо или группа лиц, отвечающих за выработку требований к БД, ее проектирование, создание, эффективное использование и сопровождение. В процессе эксплуатации АБД обычно следит за функционированием информационной системы, обеспечивает защиту от несанкционированного доступа, контролирует избыточность, непротиворечивость, сохранность и достоверность хранимой в БД информации. Для однопользовательских информационных систем функции АБД обычно возлагаются на лиц, непосредственно работающих с приложением БД.

Вычислительной сети АБД, как правило, взаимодействует с администратором сети. В обязанности последнего входят контроль за функционированием аппаратно-программных средств

сети, реконфигурация сети, восстановление программного обеспечения после сбоев и отказов оборудования, профилактические мероприятия и обеспечение разграничения доступа.

Вычислительная система (ВС) представляет собой совокупность взаимосвязанных и согласованно действующих ЭВМ или процессоров и других устройств, обеспечивающих автоматизацию процессов приема, обработки и выдачи информации потребителям. Поскольку основными функциями БнД являются хранение и обработка данных, то используемая ВС, наряду с приемлемой мощностью центральных процессоров (ЦП) должна иметь достаточный объем оперативной и внешней памяти прямого доступа.

Обслуживающий персонал выполняет функции поддержания технических и программных средств в работоспособном состоянии. Он проводит профилактические, регламентные, восстановительные и другие работы по планам, а также по мере необходимости.

2. Файловые системы хранения данных.

Фа́йловая систе́ма (англ. file system) — порядок, определяющий способ организации, хранения и именования данных на носителях информации в компьютерах, а также в другом электронном оборудовании: цифровых фотоаппаратах, мобильных телефонах и т. п. Файловая система определяет формат содержимого и способ физического хранения информации, которую принято группировать в виде файлов. Конкретная файловая система определяет размер имен файлов и (каталогов), максимальный возможный размер файла и раздела, набор атрибутов файла. Некоторые файловые системы предоставляют сервисные возможности, например, разграничение доступа или шифрование файлов.

Файловая система связывает носитель информации с одной стороны и API для доступа к файлам — с другой. Когда прикладная программа обращается к файлу, она не имеет никакого представления о том, каким образом расположена информация в конкретном файле, так же, как и на каком физическом типе носителя (CD, жёстком диске, магнитной ленте, блоке флеш-памяти или другом) он записан. Всё, что знает программа — это имя файла, его размер и атрибуты. Эти данные она получает от драйвера файловой системы. Именно файловая система устанавливает, где и как будет записан файл на физическом носителе (например, жёстком диске).

С точки зрения операционной системы (ОС), весь диск представляет собой набор кластеров (как правило, размером 512 байт и больше). Драйверы файловой системы организуют кластеры в файлы и каталоги (реально являющиеся файлами, содержащими список файлов в этом каталоге). Эти же драйверы отслеживают, какие из кластеров в настоящее время используются, какие свободны, какие помечены как неисправные.

Однако файловая система не обязательно напрямую связана с физическим носителем информации. Существуют виртуальные файловые системы, а также сетевые файловые системы, которые являются лишь способом доступа к файлам, находящимся на удалённом компьютере.

Практически всегда файлы на дисках объединяются в каталоги.

В простейшем случае все файлы на данном диске хранятся в одном каталоге. Такая одноуровневая схема использовалась в CP/M и в первой версии MS-DOS 1.0. Иерархическая файловая система со вложенными друг в друга каталогами впервые появилась в Multics, затем в UNIX.

Wiki.txt
Tornado.jpg
Notepad.exe
(Одноуровневая файловая система)

Каталоги на разных дисках могут образовывать несколько отдельных деревьев, как в DOS/Windows, или же объединяться в одно дерево, общее для всех дисков, как в UNIX-подобных системах.

C:
\Program files
\CDEx

```
        \CDEx.exe
        \CDEx.hlp
        \mppenc.exe
  \Мои документы
    \Wiki.txt
    \Tornado.jpg
D:
  \Music
    \ABBA
      \1974 Waterloo
      \1976 Arrival
        \Money, Money, Money.ogg
      \1977 The Album
```

(Иерархическая файловая система Windows/DOS)

В UNIX существует только один корневой каталог, а все остальные файлы и каталоги вложены в него. Чтобы получить доступ к файлам и каталогам на каком-нибудь диске, необходимо смонтировать этот диск командой mount. Например, чтобы открыть файлы на CD, нужно, говоря простым языком, сказать операционной системе: «возьми файловую систему на этом компакт-диске и покажи её в каталоге /mnt/cdrom». Все файлы и каталоги, находящиеся на CD, появятся в этом каталоге /mnt/cdrom, который называется точкой монтирования (англ. mount point). В большинстве UNIX-подобных систем съёмные диски (дискеты и CD), флеш-накопители и другие внешние устройства хранения данных монтируют в каталог /mnt, /mount или /media. Unix и UNIX-подобные операционные системы также позволяют автоматически монтировать диски при загрузке операционной системы.

```
/
  /usr
    /bin
      /arch
      /ls
      /raw
    /lib
      /libhistory.so.5.2
      /libgpm.so.1
  /home
    /lost+found
      /host.sh
    /guest
      /Pictures
        /example.png
      /Video
        /matrix.avi
        /news
          /lost_ship.mpeg
```

(Иерархическая файловая система в Unix и UNIX-подобных операционных системах)

Обратите внимание на использование слешей в файловых системах Windows, UNIX и UNIX-подобных операционных системах (В Windows используется обратный слеш «\», а в UNIX и UNIX-подобных операционных системах простой слеш «/»)

Кроме того, следует отметить, что вышеописанная система позволяет монтировать не только файловые системы физических устройств, но и отдельные каталоги (параметр --bind) или, например, образ ISO (опция loop). Такие надстройки, как FUSE, позволяют также монтировать, например, целый каталог на FTP и ещё очень большое количество различных ресурсов.

Ещё более сложная структура применяется в NTFS и HFS. В этих файловых системах каждый файл представляет собой набор атрибутов. Атрибутами считаются не только традиционные только для чтения, системный, но и имя файла, размер и даже содержимое. Таким образом, для NTFS и HFS то, что хранится в файле, — это всего лишь один из его атрибутов.

Если следовать этой логике, один файл может содержать несколько вариантов содержимого. Таким образом, в одном файле можно хранить несколько версий одного документа, а также дополнительные данные (значок файла, связанная с файлом программа). Такая организация типична для HFS на Macintosh.

По предназначению файловые системы можно классифицировать на нижеследующие категории:

- Для носителей с произвольным доступом (например, жёсткий диск): FAT32, HPFS, ext2 и др. Поскольку доступ к дискам в несколько раз медленнее, чем доступ к оперативной памяти, для прироста производительности во многих файловых системах применяется асинхронная запись изменений на диск. Для этого применяется либо журналирование, например в ext3, ReiserFS, JFS, NTFS, XFS, либо механизм soft updates и др. Журналирование широко распространено в Linux, применяется в NTFS. Soft updates — в BSD системах.
- Для носителей с последовательным доступом (например, магнитные ленты): QIC и др.
- Для оптических носителей — CD и DVD: ISO9660, HFS, UDF и др.
- Виртуальные файловые системы: AEFS и др.
- Сетевые файловые системы: NFS, CIFS, SSHFS, GmailFS и др.
- Для флэш-памяти: YAFFS, ExtremeFFS, exFAT.

Немного выпадают из общей классификации специализированные файловые системы: ZFS (собственно файловой системой является только часть ZFS), VMFS (т. н. кластерная файловая система, которая предназначена для хранения других файловых систем) и др.

Основные функции любой файловой системы нацелены на решение следующих задач:

- именование файлов;
- программный интерфейс работы с файлами для приложений;
- отображения логической модели файловой системы на физическую организацию хранилища данных;
- организация устойчивости файловой системы к сбоям питания, ошибкам аппаратных и программных средств;
- содержание параметров файла, необходимых для правильного его взаимодействия с другими объектами системы (ядро, приложения и пр.).

В многопользовательских системах появляется ещё одна задача: защита файлов одного пользователя от несанкционированного доступа другого пользователя, а также обеспечение совместной работы с файлами, к примеру, при открытии файла одним из пользователей, для других этот же файл временно будет доступен в режиме «только чтение».

3. Области применения баз данных.

Автоматизированные информационные системы (АИС), основу которых составляют базы данных, появились в 60-х годах XX века в военной промышленности и бизнесе — там, где были накоплены значительные объемы полезных данных. Первоначально АИС были ориентированы лишь на работу с информацией фактического характера — числовыми или текстовыми характеристиками объектов. Затем по мере развития техники появилась возможность обработки текстовой информации на естественном языке.

Принципы хранения разных видов информации в АИС аналогичны, но алгоритмы ее обработки определяются характером ин формационных ресурсов. Соответственно различают два класса АИС: документальные и фактографические.

Документальные АИС служат для работы с документами на естественном языке. Наиболее распространенный тип документальных АИС — информационно-поисковые системы, предназначенные для накопления и подбора документов, удовлетворяющих

заданным критериям. Эти системы могут выполнять просмотр и подборку монографий, публикаций в периодике, сообщений пресс- агентств, текстов законодательных актов и т.д.

Фактографические АИС оперируют фактическими сведениями, представленными в формализованном виде, и используются для решения задач обработки данных.

Обработка данных — специальный класс решаемых на ЭВМ задач, связанных с вводом, хранением, сортировкой, отбором и группировкой записей данных однородной структуры. К задачам этого класса относятся: учет товаров в магазинах и на складах; начисление зарплаты; управление производством, финансами, телекоммуникациями и т. п.

Различают фактографические АИС оперативной обработки данных, подразумевающие быстрое обслуживание относительно простых запросов от большого числа пользователей, и фактографические АИС аналитической обработки, ориентированные на выполнение сложных запросов, требующих проведения статистической обработки исторических (накопленных за некоторый промежуток времени) данных, моделирования процессов предметной области и прогнозирования развития этих процессов.

Таким образом, АИС применяются в следующих областях:

- организация хранилищ данных;
- системы анализа данных;
- системы принятия решений;
- мобильные и персональные базы данных;
- географические базы данных;
- мультимедиа базы данных;
- распределенные информационные системы.

Каждая информационная система в зависимости от назначения имеет дело с той или иной частью конкретного мира, которую принято называть ее предметной областью. Анализ предмет ной области является необходимым начальным этапом разработки любой информационной системы. Именно на этом этапе определяются информационные потребности всей совокупности пользователей будущей системы, которые, в свою очередь, предопределяют содержание ее базы данных. Предметная область конкретной информационной системы рассматривается, прежде все го, как некоторая совокупность реальных объектов, которые представляют интерес для ее пользователей. Примерами объектов предметной области могут служить персональные ЭВМ, программные продукты и их пользователи. Каждый из этих объектов обладает определенным набором свойств (атрибутов). Так, например, компьютер характеризуется названием фирмы-производителя, идёнтификатором модели, типом микропроцессора, объемом оперативной и внешней памяти, типом графической карты и т. д.

Информационный объект это описание некоторой сущности предметной области, т. е. реального объекта, процесса, явления или события. Информационный объект (сущность) образуется совокупностью логически взаимосвязанных атрибутов (свойств), представляющих собой качественные и количественные характеристики объекта (сущности).

Между объектами предметной области могут существовать связи, имеющие различный содержательный смысл. Эти связи могут быть обязательными или факультативными (необязательными).

Если вновь порожденный объект оказывается по необходимости связанным с каким-либо объектом предметной области, то между этими двумя объектами существует обязательная связь. В противном случае связь является факультативной. Например, обязательная связь Замещает существует между двумя объектами СОТРУДНИК и ДОЛЖНОСТЬ в предметной области кадровой информационной системы, т. е. каждый принимаемый в организацию сотрудник зачисляется на какую-либо должность и не может быть сотрудника, не замещающего какой-либо должности. В то же время связь Замещает между типами объектов СОТРУДНИК и ДОЛЖНОСТЬ является факультативной, поскольку могут существовать вакантные должности. Совокупность объектов предметной области и связей между ними характеризует структуру предметной области. Множество объектов

предметной области, значения атрибутов объектов и связи между ними могут изменяться во времени. Изменения могут сводиться к появлению новых или исключению из рассмотрения некоторых существующих объектов в предметной области, установлению новых или разрушению существующих связей между ними. Следовательно, с каждым моментом времени можно сопоставить некоторое состояние предметной области.

Информационно-логическая модель (ИЛМ) — это совокупность информационных объектов (сущностей) предметной области и связей между ними. Процесс создания информационной модели начинается с определения концептуальных требований будущих пользователей БД. Требования отдельных пользователей интегрируются в едином обобщенном представлении, которое называют концептуальной моделью данной предметной области.

Такая модель отображает предметную область в виде взаимосвязанных объектов без указания способов их физического хранения.

Концептуальная модель представляет собой интегрированные концептуальные требования всех пользователей к базе данных данной предметной области. При этом усилия разработчика должны быть направлены в основном на структуризацию данных, принадлежащих будущим пользователям БД и выявление взаимосвязей между ними.

Возможно, что отраженные в концептуальной модели взаимосвязи между объектами окажутся впоследствии нереализуемыми средствами выбранной СУБд, что потребует ее изменения. Вер сия концептуальной модели, которая может быть реализована конкретной СУБд, называется логической моделью.

Логическая модель, отражающая логические связи между атрибутами объектов вне зависимости от их содержания и среды хранения, может быть реляционной, иерархической или сете вой. Таким образом, логическая модель отображает логические связи между информационными данными в данной концептуальной модели.

Различным пользователям в информационной модели соответствуют различные подмножества ее логической модели, которые называются внешними моделями пользователей. Таким образом, внешняя модель пользователя представляет собой отображение его концептуальных требований в логической модели и соответствует тем представлениям, которые этот пользователь получает о предметной области на основе логической модели. Следовательно, насколько хорошо спроектирована внешняя модель, настолько полно и точно информационная модель отображает предметную область и настолько полно и точно работает автоматизированная система управления этой предмет ной областью.

Логическая модель отображается в физическую память, которая может быть построена на электронных, магнитных, оптических, биологических или других принципах.

Внутренняя модель предметной области определяет размещение данных, методы доступа к ним и технику индексирования в данной логической модели и иначе называется физической моделью. Информационные данные любого пользователя в БД должны быть независимы от всех других пользователей, т. е. не должны оказывать влияния на существующие внешние модели. Это положение отражает первый уровень независимости данных. С другой стороны, внешние модели пользователей никак не связаны с типом физической памяти, в которой будут храниться данные, и с физическими методами доступа к этим данным. Это положение отражает второй уровень независимости данных.

1. 3 Лекция №3 (2 часа).

Тема: «Топология баз данных»

1.3.1 Вопросы лекции:

1. Основные типы структур данных
2. Информационная модель данных и ее состав
3. Классификация баз данных

1.3.2 Краткое содержание вопросов:

1. Основные типы структур данных.

Необходимым условием хранения информации в памяти компьютера является возможность преобразования этой самой информации в подходящую для компьютера форму. В том случае, если это условие выполняется, следует определить структуру, пригодную именно для наличествующей информации, ту, которая предоставит требующийся набор возможностей работы с ней. Здесь под структурой понимается способ представления информации, посредством которого совокупность отдельно взятых элементов образует нечто единое, обусловленное их взаимосвязью друг с другом. Скомпонованные по каким-либо правилам и логически связанные межу собой, данные могут весьма эффективно обрабатываться, так как общая для них структура предоставляет набор возможностей управления ими – одно из того за счет чего достигаются высокие результаты в решениях тех или иных задач. Но не каждый объект представляем в произвольной форме, а возможно и вовсе для него имеется лишь один единственный метод интерпретации, следовательно, несомненным плюсом для программиста будет знание всех существующих структур данных. Таким образом, часто приходиться делать выбор между различными методами хранения информации, и от такого выбора зависит работоспособность продукта.

Говоря о не вычислительной технике, можно показать ни один случай, где у информации видна явная структура. Наглядным примером служат книги самого разного содержания. Они разбиты на страницы, параграфы и главы, имеют, как правило, оглавление, то есть интерфейс пользования ими. В широком смысле, структурой обладает всякое живое существо, без нее органика не смогла бы существовать.

Вполне вероятно, читателю приходилось сталкиваться со структурами данных непосредственно в информатике, например, с теми, что встроены в язык программирования. Часто они именуются типами данных. К таковым относятся: массивы, числа, строки, файлы др.

Методы хранения информации, называемые «простыми», т. е. неделимыми на составные части, предпочтительнее изучать вместе с конкретным языком программирования, либо же глубоко углубляться в суть их работы. Поэтому здесь будут рассмотрены лишь «интегрированные» структуры, те которые состоят из простых, а именно: массивы, списки, деревья и графы.

Массивы

Массив – это структура данных с фиксированным и упорядоченным набором однотипных элементов (компонентов). Доступ к какому-либо из элементов массива осуществляется по имени и номеру (индексу) этого элемента. Количество индексов определяет размерность массива. Так, например, чаще всего встречаются одномерные (вектора) и двумерные (матрицы) массивы. Первые имеют один индекс, вторые – два. Пусть одномерный массив называется A, тогда для получения доступа к его i-ому элементу потребуется указать название массива и номер требуемого элемента: A[i]. Когда A – матрица, то она представляема в виде таблицы, доступ к элементам которой осуществляется по имени массива, а также номерам строки и столбца, на пересечении которых расположен элемент: A[i, j], где i – номер строки, j – номер столбца.

В разных языках программирования работа с массивами может в чем-то различаться, но основные принципы, как правило, везде одни. В языке Pascal, обращение к одномерному и

двумерному массиву происходит точно так, как это показано выше, а, например, в C++ двумерный массив следует указывать так: A[i][j]. Элементы массива нумеруются поочередно. На то, с какого значения начинается нумерация, влияет язык программирования. Чаще всего этим значением является 0 или 1.

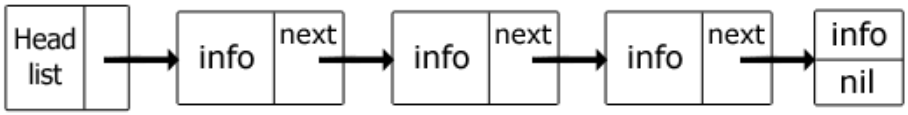
Массивы, описанного типа называются статическими, но существуют также массивы по определенным признакам отличные от них: динамические и гетерогенные. Динамичность первых характеризуется непостоянностью размера, т. е. по мере выполнения программы размер динамического массива может изменяться. Такая функция делает работу с данными более гибкой, но при этом приходится жертвовать быстродействием, да и сам процесс усложняется. Обязательный критерий статического массива, как было сказано, это однородность данных, единовременно хранящихся в нем. Когда же данное условие не выполняется, то массив является гетерогенным. Его использование обусловлено недостатками, которые имеются в предыдущем виде, но оно оправданно во многих случаях.

Таким образом, даже если Вы определились со структурой, и в качестве нее выбрали массив, то этого все же недостаточно. Ведь массив это только общее обозначение, род для некоторого числа возможных реализаций. Поэтому необходимо определиться с конкретным способом представления, с наиболее подходящим массивом.

Списки

Список – абстрактный тип данных, реализующий упорядоченный набор значений. Списки отличаются от массивов тем, что доступ к их элементам осуществляется последовательно, в то время как массивы – структура данных произвольного доступа. Данный абстрактный тип имеет несколько реализаций в виде структур данных. Некоторые из них будут рассмотрены здесь.

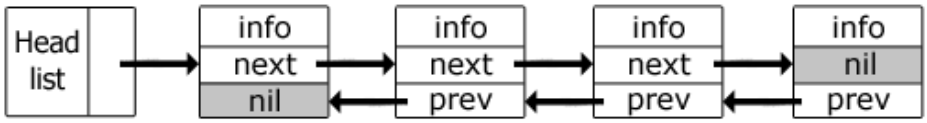
Список (связный список) – это структура данных, представляющая собой конечное множество упорядоченных элементов, связанных друг с другом посредствам указателей. Каждый элемент структуры содержит поле с какой-либо информацией, а также указатель на следующий элемент. В отличие от массива, к элементам списка нет произвольного доступа.



Односвязный список

В односвязном списке, приведенным выше, начальным элементом является Head list (голова списка [произвольное наименование]), а все остальное называется хвостом. Хвост списка составляют элементы, разделенные на две части: информационную (поле info) и указательную (поле next). В последнем элементе вместо указателя, содержится признак конца списка – nil.

Односвязный список не слишком удобен, т. к. из одной точки есть возможность попасть лишь в следующую точку, двигаясь тем самым в конец. Когда кроме указателя на следующий элемент есть указатель и на предыдущий, то такой список называется двусвязным.

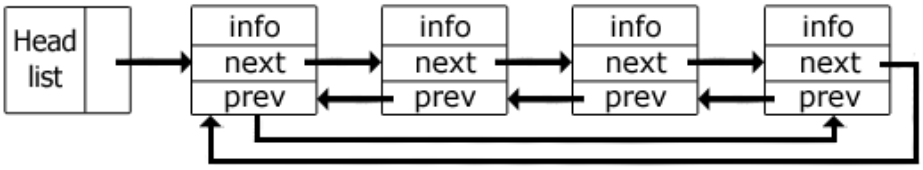


Двусвязный список

Возможность двигаться как вперед, так и назад полезна для выполнения некоторых операций, но дополнительные указатели требуют задействования большего количества памяти, чем таковой необходимо в эквивалентном односвязном списке.

Для двух видов списков описанных выше существует подвид, называемый кольцевым списком. Сделать из односвязного списка кольцевой можно добавив всего лишь один

указатель в последний элемент, так чтобы он ссылался на первый. А для двусвязного потребуется два указателя: на первый и последний элементы.

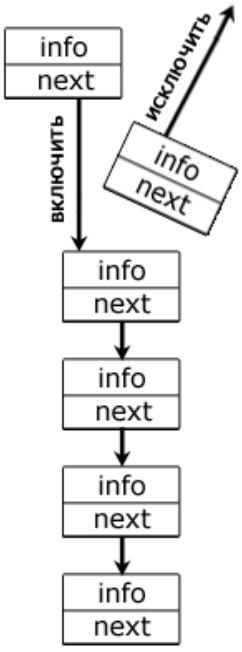


Кольцевой список

Помимо рассмотренных видов списочных структур есть и другие способы организации данных по типу «список», но они, как правило, во многом схожи с разобранными, поэтому здесь они будут опущены.

Кроме различия по связям, списки делятся по методам работы с данными. О некоторых таких методах сказано далее.

Стек

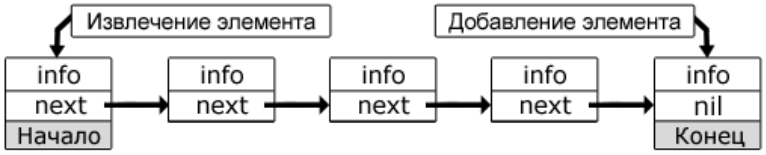


Стек характерен тем, что получить доступ к его элементом можно лишь с одного конца, называемого вершиной стека, иначе говоря: стек – структура данных, функционирующая по принципу LIFO (last in — first out, «последним пришёл — первым вышел»). Изобразить эту структуру данных лучше в виде вертикального списка, например, стопки каких-либо вещей, где чтобы воспользоваться одной из них нужно поднять все те вещи, что лежат выше нее, а положить предмет можно лишь на вверх стопки.

В показанном односвязном списке операции над элементами происходят строго с одного конца: для включения нужного элемента в пятую по счету ячейку необходимо исключить тот элемент, который занимает эту позицию. Если бы было, например 6 элементов, а вставить конкретный элемент требовалось также в пятую ячейку, то исключить бы пришлось уже два элемента.

Очередь

Структура данных «Очередь» использует принцип организации FIFO (First In, First Out — «первым пришёл — первым вышел»). В некотором смысле такой метод более справедлив, чем тот, по которому функционирует стек, ведь простое правило, лежащее в основе привычных очередей в различные магазины, больницы считается вполне справедливым, а именно оно является базисом этой структуры. Пусть данное наблюдение будет примером. Строго говоря, очередь – это список, добавление элементов в который допустимо, лишь в его конец, а их извлечение производиться с другой стороны, называемой началом списка.



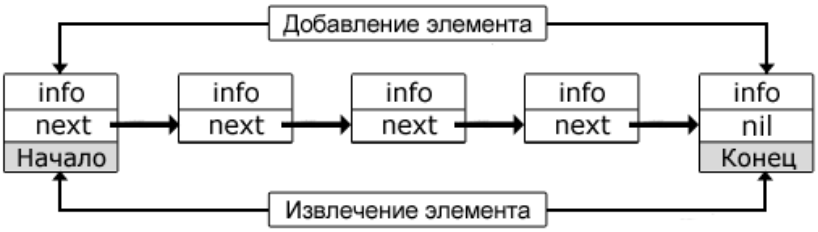
Очередь

Эта структура одновременно функционирует по двум способам организации данных: FIFO и LIFO. Поэтому ее допустимо отнести к отдельной программной единице, полученной в результате суммирования двух предыдущих видов списка.

Дек

Дек (deque — double ended queue, «двухсторонняя очередь») – стек с двумя концами. Действительно, несмотря конкретный перевод, дек можно определять не только как двухстороннюю очередь, но и как стек, имеющий два конца. Это означает, что данный вид

списка позволяет добавлять элементы в начало и в конец, и то же самое справедливо для операции извлечения.



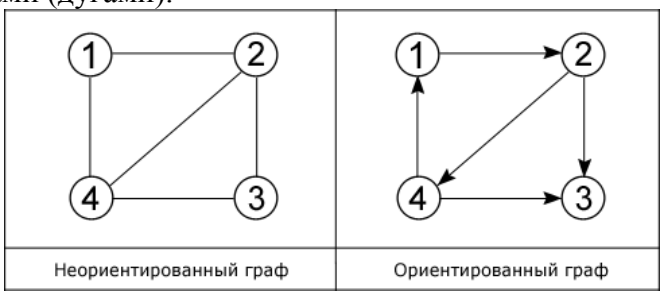
Дек

Эта структура одновременно работает по двум способам организации данных: FIFO и LIFO. Поэтому ее допустимо отнести к отдельной программной единице, полученной в результате суммирования двух предыдущих видов списка.

Графы

Раздел дискретной математики, занимающийся изучением графов, называется теорией графов. В теории графов подробно рассматриваются известные понятия, свойства, способы представления и области применения этих математических объектов. Нас же интересует, лишь те ее аспекты, которые важны в программировании.

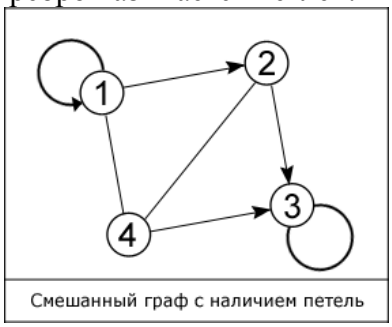
Граф – совокупность точек, соединенных линиями. Точки называются вершинами (узлами), а линии – ребрами (дугами).



Как показано на рисунке различают два основных вида графов: ориентированные и неориентированные. В первых ребра являются направленными, т. е. существует только одно доступное направление между двумя связными вершинами, например из вершины 1 можно пройти в вершину 2, но не наоборот. В неориентированном связном графе из каждой вершины можно пройти в каждую и обратно. Частный случай двух этих видов – смешанный граф. Он характерен наличием как ориентированных, так и неориентированных ребер.

Степень входа вершины – количество входящих в нее ребер, степень выхода – количество исходящих ребер.

Ребра графа необязательно должны быть прямыми, а вершины обозначаться именно цифрами, так как показано на рисунке. К тому же встречаются такие графы, ребрам которых поставлено в соответствие конкретное значение, они именуются взвешенными графами, а это значение – весом ребра. Когда у ребра оба конца совпадают, т. е. ребро выходит из вершины F и входит в нее, то такое ребро называется петлей.

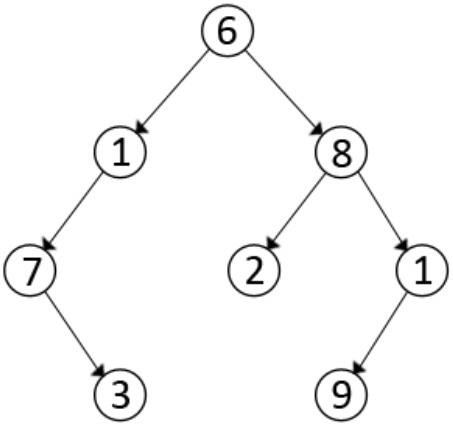


Графы широко используются в структурах, созданных человеком, например в компьютерных и транспортных сетях, web-технологиях. Специальные способы

представления позволяют использовать граф в информатике (в вычислительных машинах). Самые известные из них: «Матрица смежности», «Матрица инцидентности», «Список смежности», «Список рёбер». Два первых, как понятно из названия, для репрезентации графа используют матрицу, а два последних – список.

Деревья

Дерево – структура данных



Неупорядоченное дерево

Дерево как математический объект это абстракция из соименных единиц, встречающихся в природе. Схожесть структуры естественных деревьев с графами определенного вида говорит о допущении установления аналогии между ними. А именно со связанными и вместе с этим ациклическими (не имеющими циклов) графами. Последние по своему строению действительно напоминают деревья, но в чем то и имеются различия, например, принято изображать математические деревья с корнем расположенным вверху, т. е. все ветви «растут» сверху вниз. Известно же, что в природе это совсем не так.

Поскольку дерево это по своей сути граф, у него с последним многие определения совпадают, либо интуитивно схожи. Так корневой узел (вершина 6) в структуре дерева – это единственная вершина (узел), характерная отсутствием предков, т. е. такая, что на нее не ссылается ни какая другая вершина, а из самого корневого узла можно дойти до любой из имеющихся вершин дерева, что следует из свойства связности данной структуры. Узлы, не ссылающиеся ни на какие другие узлы, иначе говоря, ни имеющие потомков называются листьями (2, 3, 9), либо терминальными узлами. Элементы, расположенные между корневым узлом и листьями – промежуточные узлы (1, 1, 7, 8). Каждый узел дерева имеет только одного предка, или если он корневой, то не имеет ни одного.

Поддерево – часть дерева, включающая некоторый корневой узел и все его узлы-потомки. Так, например, на рисунке одно из поддеревьев включает корень 8 и элементы 2, 1, 9.

С деревом можно выполнять многие операции, например, находить элементы, удалять элементы и поддеревья, вставлять поддеревья, находить корневые узлы для некоторых вершин и др. Одной из важнейших операций является обход дерева. Выделяются несколько методов обхода. Наиболее популярные из них: симметричный, прямой и обратный обход. При прямом обходе узлы-предки посещаются прежде своих потомков, а в обратном обходе, соответственно, обратная ситуация. В симметричном обходе поочередно просматриваются поддеревья главного дерева.

Представление данных в рассмотренной структуре выгодно в случае наличия у информации явной иерархии. Например, работа с данными о биологических родах и видах, служебных должностях, географических объектах и т. п. требует иерархически выраженной структуры, такой как математические деревья.

2. Информационная модель данных и ее состав.

Каждая информационная система в зависимости от ее назначения имеет дело с той или иной частью конкретного мира, которую принято называть предметной областью информационной системы. Анализ предметной области является необходимым начальным этапом разработки любой информационной системы. Именно на этом этапе определяются информационные потребности всей совокупности пользователей будущей системы, которые, в свою очередь, предопределяют содержание ее базы данных. Предметная область данной информационной системы рассматривается прежде всего как некоторая совокупность реальных объектов, которые представляют интерес для ее пользователей. Примерами объектов предметной области могут служить персональные ЭВМ, программные продукты, их пользователи. Каждый из них обладает определенным набором свойств (атрибутов). Так, компьютер характеризуется названием фирмы-производителя, идентификатором модели, типом микропроцессора, объемом оперативной и внешней памяти, типом графической карты и т. д.

Информационный объект — это описание некоторой сущности предметной области — реального объекта, процесса, явления или события. Информационный объект (сущность) образуется совокупностью логически взаимосвязанных атрибутов (свойств), представ-ляющих качественные и количественные характеристики объекта (сущности).

Между объектами предметной области могут существовать связи, имеющие различный содержательный смысл. Эти связи могут быть обязательными или факультативными.

Если вновь порожденный объект оказывается по необходимости связанным с каким-либо объектом предметной области, то между этими двумя объектами существует обязательная связь. В противном случае связь является факультативной (необязательной).

Обязательная связь «ЗАМЕЩАЕТ» существует, например, между двумя объектами СОТРУДНИК и ДОЛЖНОСТЬ в предметной области кадровой информационной системы. Каждый принимаемый в организацию сотрудник зачисляется на какую-либо должность и не может быть сотрудника, не замещающего какой-либо должности. В то же время связь «ЗАМЕЩАЕТСЯ» между типами объектов СОТРУДНИК и ДОЛЖНОСТЬ является факультативной, поскольку могут существовать вакантные должности.

Совокупность объектов предметной области и связей между ними характеризует (типовую) структуру предметной области.

Множество объектов предметной области, значения атрибутов объектов и связи между ними могут изменяться во времени. Изменения могут сводиться к появлению новых или исключению из рас-смотрения некоторых существующих объектов в предметной области, установлению новых или разрушению существующих связей между ними. Поэтому с каждым моментом времени можно сопоставить некоторое состояние предметной области.

Информационно-логическая модель (ИЛМ) — совокупность ин-формационных объектов (сущностей) предметной области и связей между ними.

3. Классификация баз данных.

В силу многогранности баз данных и СУБД (комплекса технических и программных средств, для хранения, поиска, защиты и использования данных) имеется множество классификационных признаков.

Классификация БД по основным признакам приведена на рис. 1.

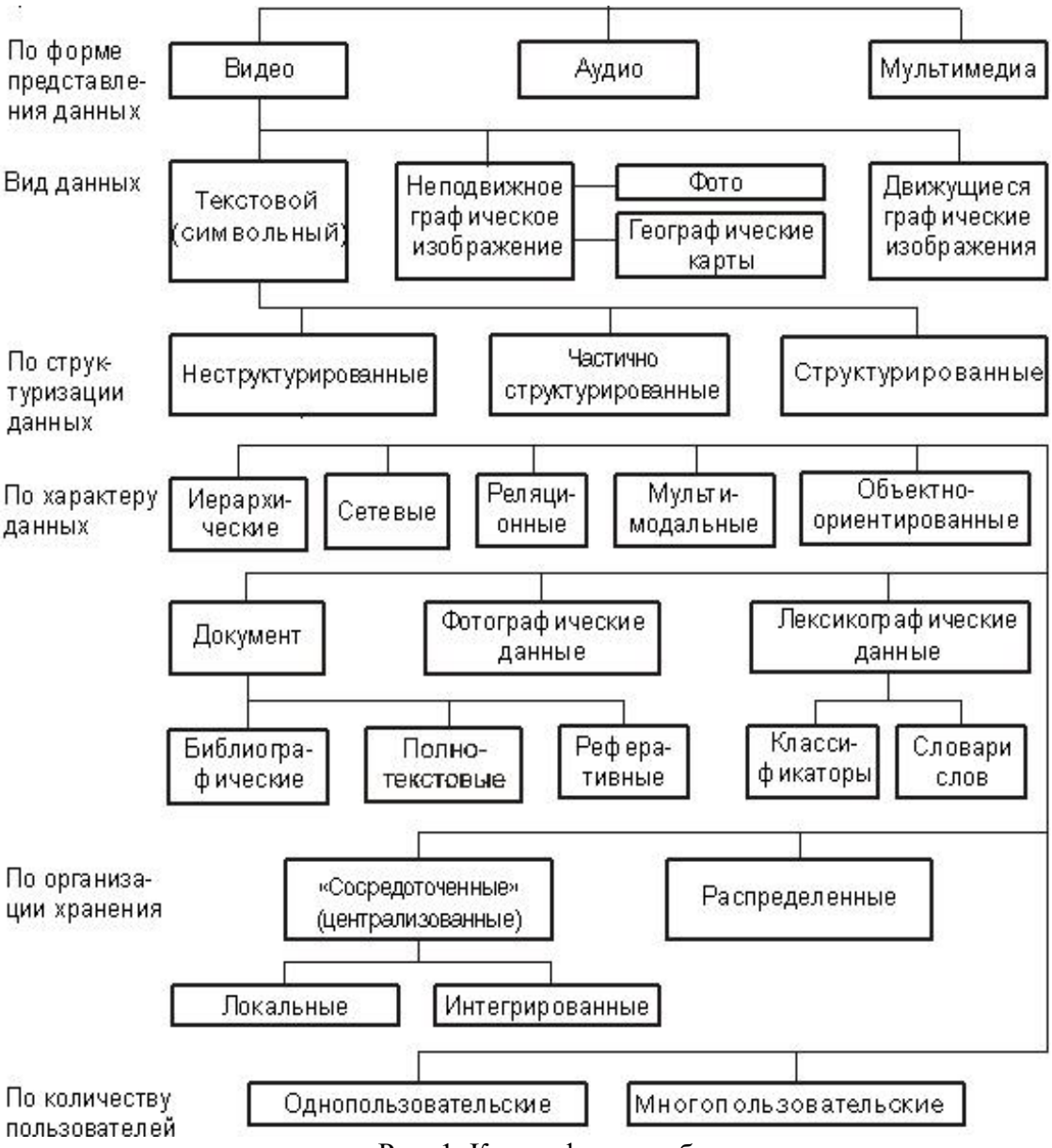


Рис. 1. Классификация баз данных

Базы данных могут классифицироваться и с точки зрения экономической: по условиям предоставления услуг - бесплатные и платные (бесприбыльные, коммерческие); по форме собственности - государственные, негосударственные; по степени доступности - общедоступные, с ограниченным кругом пользователей.

1. 4 Лекция №4 (2 часа).

Тема: «Введение в реляционную модель данных»

1.4.1 Вопросы лекции:

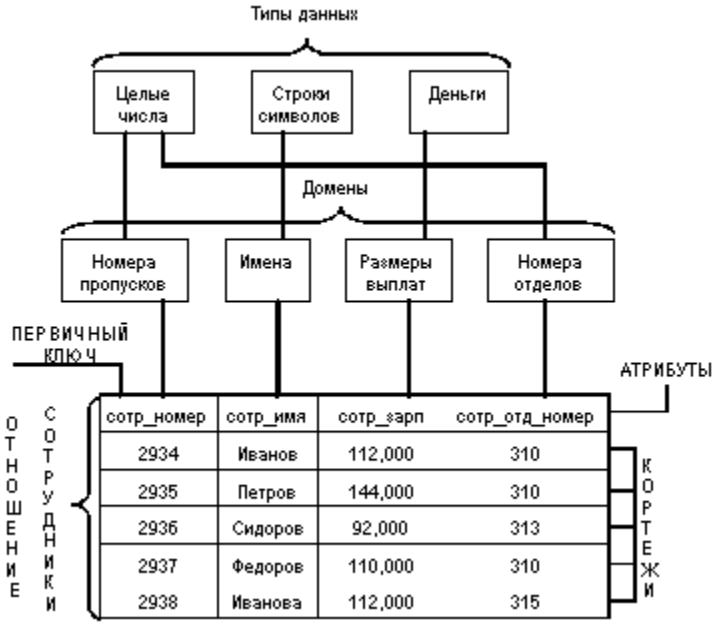
1. Основные понятия реляционных баз данных
2. Фундаментальные свойства отношений
3. Реляционная модель данных

1.4.2 Краткое содержание вопросов:

1. Основные понятия реляционных баз данных.

Основными понятиями реляционных баз данных являются тип данных, домен, атрибут, кортеж, первичный ключ и отношение.

Для начала покажем смысл этих понятий на примере отношения СОТРУДНИКИ, содержащего информацию о сотрудниках некоторой организации:



сотр_номер	сотр_имя	сотр_зарп	сотр_отд_номер
2934	Иванов	112,000	310
2935	Петров	144,000	310
2936	Сидоров	92,000	313
2937	Федоров	110,000	310
2938	Иванова	112,000	315

Понятие **тип данных** в реляционной модели данных полностью адекватно понятию типа данных в языках программирования. Обычно в современных реляционных БД допускается хранение символьных, числовых данных, битовых строк, специализированных числовых данных (таких как "деньги"), а также специальных "темпоральных" данных (дата, время, временной интервал). Достаточно активно развивается подход к расширению возможностей реляционных систем абстрактными типами данных (соответствующими возможностями обладают, например, системы семейства Ingres/Postgres). В нашем примере мы имеем дело с данными трех типов: строки символов, целые числа и "деньги".

Понятие **домена** более специфично для баз данных, хотя и имеет некоторые аналогии с подтипами в некоторых языках программирования. В самом общем виде домен определяется заданием некоторого базового типа данных, к которому относятся элементы домена, и произвольного логического выражения, применяемого к элементу типа данных. Если вычисление этого логического выражения дает результат "истина", то элемент данных является элементом домена.

Наиболее правильной интуитивной трактовкой понятия домена является понимание домена как допустимого потенциального множества значений данного типа. Например, домен "Имена" в нашем примере определен на базовом типе строк символов, но в число его

значений могут входить только те строки, которые могут изображать имя (в частности, такие строки не могут начинаться с мягкого знака).

Следует отметить также семантическую нагрузку понятия домена: данные считаются сравнимыми только в том случае, когда они относятся к одному домену. В нашем примере значения доменов "Номера пропусков" и "Номера групп" относятся к типу целых чисел, но не являются сравнимыми. Заметим, что в большинстве реляционных СУБД понятие домена не используется, хотя в Oracle V.7 оно уже поддерживается.

Схема отношения - это именованное множество пар {имя атрибута, имя домена (или типа, если понятие домена не поддерживается)}. Степень или "арность" схемы отношения - мощность этого множества. Степень отношения СОТРУДНИКИ равна четырем, то есть оно является 4-арным. Если все атрибуты одного отношения определены на разных доменах, осмысленно использовать для именования атрибутов имена соответствующих доменов (не забывая, конечно, о том, что это является всего лишь удобным способом именования и не устраняет различия между понятиями домена и атрибута).

Схема БД (в структурном смысле) - это набор именованных схем отношений.

Кортеж, соответствующий данной схеме отношения, - это множество пар {имя атрибута, значение}, которое содержит одно вхождение каждого имени атрибута, принадлежащего схеме отношения. "Значение" является допустимым значением домена данного атрибута (или типа данных, если понятие домена не поддерживается). Тем самым, степень или "арность" кортежа, т.е. число элементов в нем, совпадает с "арностью" соответствующей схемы отношения. Попросту говоря, кортеж - это набор именованных значений заданного типа.

Отношение - это множество кортежей, соответствующих одной схеме отношения. Иногда, чтобы не путаться, говорят "отношение-схема" и "отношение-экземпляр", иногда схему отношения называют заголовком отношения, а отношение как набор кортежей - телом отношения. На самом деле, понятие схемы отношения ближе всего к понятию структурного типа данных в языках программирования. Было бы вполне логично разрешать отдельно определять схему отношения, а затем одно или несколько отношений с данной схемой.

Однако в реляционных базах данных это не принято. Имя схемы отношения в таких базах данных всегда совпадает с именем соответствующего отношения-экземпляра. В классических реляционных базах данных после определения схемы базы данных изменяются только отношения-экземпляры. В них могут появляться новые и удаляться или модифицироваться существующие кортежи. Однако во многих реализациях допускается и изменение схемы базы данных: определение новых и изменение существующих схем отношения. Это принято называть эволюцией схемы базы данных.

Обычным житейским представлением отношения является таблица, заголовком которой является схема отношения, а строками - кортежи отношения-экземпляра; в этом случае имена атрибутов именуют столбцы этой таблицы. Поэтому иногда говорят "столбец таблицы", имея в виду "атрибут отношения". Когда мы перейдем к рассмотрению практических вопросов организации реляционных баз данных и средств управления, мы будем использовать эту житейскую терминологию. Этой терминологии придерживаются в большинстве коммерческих реляционных СУБД.

Реляционная база данных - это набор отношений, имена которых совпадают с именами схем отношений в схеме БД.

Как видно, основные структурные понятия реляционной модели данных (если не считать понятия домена) имеют очень простую интуитивную интерпретацию, хотя в теории реляционных БД все они определяются абсолютно формально и точно.

2. Фундаментальные свойства отношений.

Остановимся теперь на некоторых важных свойствах отношений, которые следуют из приведенных ранее определений:

Отсутствие кортежей-дубликатов

То свойство, что отношения не содержат кортежей-дубликатов, следует из определения отношения как множества кортежей. В классической теории множеств по определению каждое множество состоит из различных элементов.

Из этого свойства вытекает наличие у каждого отношения так называемого первичного ключа - набора атрибутов, значения которых однозначно определяют кортеж отношения. Для каждого отношения по крайней мере полный набор его атрибутов обладает этим свойством. Однако при формальном определении первичного ключа требуется обеспечение его "минимальности", т.е. в набор атрибутов первичного ключа не должны входить такие атрибуты, которые можно отбросить без ущерба для основного свойства - однозначно определять кортеж. Понятие первичного ключа является исключительно важным в связи с понятием целостности баз данных. Во многих практических реализациях РСУБД допускается нарушение свойства уникальности кортежей для промежуточных отношений, порождаемых неявно при выполнении запросов. Такие отношения являются не множествами, а мультимножествами, что в ряде случаев позволяет добиться определенных преимуществ, но иногда приводит к серьезным проблемам.

Отсутствие упорядоченности кортежей

Свойство отсутствия упорядоченности кортежей отношения также является следствием определения отношения-экземпляра как множества кортежей. Отсутствие требования к поддержанию порядка на множестве кортежей отношения дает дополнительную гибкость СУБД при хранении баз данных во внешней памяти и при выполнении запросов к базе данных. Это не противоречит тому, что при формулировании запроса к БД, например, на языке SQL можно потребовать сортировки результирующей таблицы в соответствии со значениями некоторых столбцов. Такой результат, вообще говоря, не отношение, а некоторый упорядоченный список кортежей.

Отсутствие упорядоченности атрибутов

Атрибуты отношений не упорядочены, поскольку по определению схема отношения есть множество пар {имя атрибута, имя домена}. Для ссылки на значение атрибута в кортеже отношения всегда используется имя атрибута. Это свойство теоретически позволяет, например, модифицировать схемы существующих отношений не только путем добавления новых атрибутов, но и путем удаления существующих атрибутов. Однако в большинстве существующих систем такая возможность не допускается, и хотя упорядоченность набора атрибутов отношения явно не требуется, часто в качестве неявного порядка атрибутов используется их порядок в линейной форме определения схемы отношения.

Атомарность значений атрибутов

Значения всех атрибутов являются атомарными. Это следует из определения домена как потенциального множества значений простого типа данных, т.е. среди значений домена не могут содержаться множества значений (отношения). Принято говорить, что в реляционных базах данных допускаются только нормализованные отношения или отношения, представленные в первой нормальной форме. Потенциальным примером ненормализованного отношения является следующее:

НОМЕР_ОТДЕЛА	ОТДЕЛ		
	СОТР_НОМЕР	СОТР_ИМЯ	СОТР_ЗАРП
310	2934	Иванов	112,000
	2935	Петров	112,500
313	2937	Федоров	110,000
315	2938	Иванова	112,000

Можно сказать, что здесь мы имеем бинарное отношение, значениями атрибута ОТДЕЛЫ которого являются отношения. Заметим, что исходное отношение СОТРУДНИКИ является нормализованным вариантом отношения ОТДЕЛЫ:

СОТР_НОМЕР	СОТР_ИМЯ	СОТР_ЗАРП	СОТР_ОТД_НОМЕР
2934	Иванов	112,000	310
2935	Петров	144,000	310
2936	Сидоров	92,000	313
2937	Федоров	110,000	310
2938	Иванова	112,000	315

Нормализованные отношения составляют основу классического реляционного подхода к организации баз данных. Они обладают некоторыми ограничениями (не любую информацию удобно представлять в виде плоских таблиц), но существенно упрощают манипулирование данными. Рассмотрим, например, два идентичных оператора занесения кортежа:

Зачислить сотрудника Кузнецова (пропуск номер 3000, зарплата 115,000) в отдел номер 320 и

Зачислить сотрудника Кузнецова (пропуск номер 3000, зарплата 115,000) в отдел номер 310.

Если информация о сотрудниках представлена в виде отношения СОТРУДНИКИ, оба оператора будут выполняться одинаково (вставить кортеж в отношение СОТРУДНИКИ). Если же работать с ненормализованным отношением ОТДЕЛЫ, то первый оператор выразится в занесение кортежа, а второй - в добавление информации о Кузнецове в множественное значение атрибута ОТДЕЛ кортежа с первичным ключом 310.

3. Реляционная модель данных.

Когда в предыдущих разделах мы говорили об основных понятиях реляционных баз данных, мы не опирались на какую-либо конкретную реализацию. Эти рассуждения в равной степени относились к любой системе, при построении которой использовался реляционный подход.Другими словами, мы использовали понятия так называемой реляционной модели данных. Модель данных описывает некоторый набор родовых понятий и признаков, которыми должны обладать все конкретные СУБД и управляемые ими базы данных, если они основываются на этой модели. Наличие модели данных позволяет сравнивать конкретные реализации, используя один общий язык. Хотя понятие модели данных является общим, и можно говорить о иерархической, сетевой, некоторой семантической и т.д. моделях данных, нужно отметить, что это понятие было введено в обиход применительно к реляционным системам и наиболее эффективно используется именно в этом контексте. Попытки прямолинейного применения аналогичных моделей к дореляционным организациям показывают, что реляционная модель слишком "велика" для них, а для постреляционных организаций она оказывается "мала".

Наиболее распространенная трактовка реляционной модели данных, по-видимому, принадлежит Дейту, который воспроизводит ее (с различными уточнениями) практически во всех своих книгах. Согласно Дейту реляционная модель состоит из трех частей, описывающих разные аспекты реляционного подхода: структурной части, манипуляционной части и целостной части. В структурной части модели фиксируется, что единственной структурой данных, используемой в реляционных БД, является нормализованное n-арное отношение. По сути дела, в предыдущих двух разделах этой лекции мы рассматривали именно понятия и свойства структурной составляющей реляционной модели. В манипуляционной части модели утверждаются два фундаментальных механизма манипулирования реляционными БД - реляционная алгебра и реляционное исчисление. Первый механизм базируется в основном на классической теории множеств (с некоторыми уточнениями), а второй - на классическом логическом аппарате исчисления предикатов первого порядка. Мы рассмотрим эти механизмы более подробно на следующей лекции, а пока лишь заметим, что основной функцией манипуляционной части реляционной модели является обеспечение меры реляционности любого конкретного языка реляционных БД: язык называется

реляционным, если он обладает не меньшей выразительностью и мощностью, чем реляционная алгебра или реляционное исчисление.

Наконец, в целостной части реляционной модели данных фиксируются два базовых требования целостности, которые должны поддерживаться в любой реляционной СУБД. Первое требование называется требованием целостности сущностей. Объекту или сущности реального мира в реляционных БД соответствуют кортежи отношений. Конкретно требование состоит в том, что любой кортеж любого отношения отличим от любого другого кортежа этого отношения, т.е. другими словами, любое отношение должно обладать первичным ключом. Как мы видели в предыдущем разделе, это требование автоматически удовлетворяется, если в системе не нарушаются базовые свойства отношений.

Второе требование называется требованием целостности по ссылкам и является несколько более сложным. Очевидно, что при соблюдении нормализованности отношений сложные сущности реального мира представляются в реляционной БД в виде нескольких кортежей нескольких отношений. Например, представим, что нам требуется представить в реляционной базе данных сущность ОТДЕЛ с атрибутами ОТД_НОМЕР (номер отдела), ОТД_КОЛ (количество сотрудников) и ОТД_СОТР (набор сотрудников отдела). Для каждого сотрудника нужно хранить СОТР_НОМЕР (номер сотрудника), СОТР_ИМЯ (имя сотрудника) и СОТР_ЗАРП (заработная плата сотрудника). Как мы вскоре увидим, при правильном проектировании соответствующей БД в ней появятся два отношения: ОТДЕЛЫ (ОТД_НОМЕР, ОТД_КОЛ) (первичный ключ - ОТД_НОМЕР) и СОТРУДНИКИ (СОТР_НОМЕР, СОТР_ИМЯ, СОТР_ЗАРП, СОТР_ОТД_НОМ) (первичный ключ - СОТР_НОМЕР).

Как видно, атрибут СОТР_ОТД_НОМ появляется в отношении СОТРУДНИКИ не потому, что номер отдела является собственным свойством сотрудника, а лишь для того, чтобы иметь возможность восстановить при необходимости полную сущность ОТДЕЛ. Значение атрибута СОТР_ОТД_НОМ в любом кортеже отношения СОТРУДНИКИ должно соответствовать значению атрибута ОТД_НОМ в некотором кортеже отношения ОТДЕЛЫ. Атрибут такого рода называется внешним ключом, поскольку его значения однозначно характеризуют сущности, представленные кортежами некоторого другого отношения (т.е. задают значения их первичного ключа). Говорят, что отношение, в котором определен внешний ключ, ссылается на соответствующее отношение, в котором такой же атрибут является первичным ключом. Требование целостности по ссылкам, или требование внешнего ключа состоит в том, что для каждого значения внешнего ключа, появляющегося в ссылающемся отношении, в отношении, на которое ведет ссылка, должен найтись кортеж с таким же значением первичного ключа, либо значение внешнего ключа должно быть неопределенным (т.е. ни на что не указывать). Для нашего примера это означает, что если для сотрудника указан номер отдела, то этот отдел должен существовать.

Ограничения целостности сущности и по ссылкам должны поддерживаться СУБД. Для соблюдения целостности сущности достаточно гарантировать отсутствие в любом отношении кортежей с одним и тем же значением первичного ключа. С целостностью по ссылкам дела обстоят несколько более сложно. Понятно, что при обновлении ссылающегося отношения (вставке новых кортежей или модификации значения внешнего ключа в существующих кортежах) достаточно следить за тем, чтобы не появлялись некорректные значения внешнего ключа. Но как быть при удалении кортежа из отношения, на которое ведет ссылка? Здесь существуют три подхода, каждый из которых поддерживает целостность по ссылкам. Первый подход заключается в том, что запрещается производить удаление кортежа, на который существуют ссылки (т.е. сначала нужно либо удалить ссылающиеся кортежи, либо соответствующим образом изменить значения их внешнего ключа). При втором подходе при удалении кортежа, на который имеются ссылки, во всех ссылающихся кортежах значение внешнего ключа автоматически становится неопределенным. Наконец, третий подход (каскадное удаление) состоит в том, что при удалении кортежа из отношения, на которое ведет ссылка, из ссылающегося отношения автоматически удаляются все ссылающиеся кортежи.

В развитых реляционных СУБД обычно можно выбрать способ поддержания целостности по ссылкам для каждой отдельной ситуации определения внешнего ключа. Конечно, для принятия такого решения необходимо анализировать требования конкретной прикладной области.

1. 5 Лекция №5 (2 часа).

Тема: «Базисные средства манипулирования реляционными данными»

1.5.1 Вопросы лекции:

1. Реляционная алгебра
2. Обзор реляционной алгебры Кодда
3. Обзор реляционной алгебры А Дейта и Дарвена

1.5.2 Краткое содержание вопросов:

1. Реляционная алгебра.

Основная идея реляционной алгебры состоит в том, что коль скоро отношения являются множествами, то средства манипулирования отношениями могут базироваться на традиционных теоретико-множественных операциях, дополненных некоторыми специальными операциями, специфичными для баз данных.

Существует много подходов к определению реляционной алгебры, которые различаются набором операций и способами их интерпретации, но в принципе, более или менее равносильны. Мы опишем немного расширенный начальный вариант алгебры, который был предложен Коддом. В этом варианте набор основных алгебраических операций состоит из восьми операций, которые делятся на два класса - теоретико-множественные операции и специальные реляционные операции. В состав теоретико-множественных операций входят операции:

-объединения отношений;
-пересечения отношений;
-взятия разности отношений;
-прямого произведения отношений.

Специальные реляционные операции включают:

-ограничение отношения;
-проекцию отношения;
-соединение отношений;
-деление отношений.

Кроме того, в состав алгебры включается операция присваивания, позволяющая сохранить в базе данных результаты вычисления алгебраических выражений, и операция переименования атрибутов, дающая возможность корректно сформировать заголовок (схему) результирующего отношения.

2. Обзор реляционной алгебры Кодда.

При выполнении операции объединения (UNION) двух отношений с одинаковыми заголовками производится отношение, включающее все кортежи, которые входят хотя бы в одно из отношений-операндов.

Операция пересечения (INTERSECT) двух отношений с одинаковыми заголовками производит отношение, включающее все кортежи, которые входят в оба отношения-операнда.

Отношение, являющееся разностью (MINUS) двух отношений с одинаковыми заголовками, включает все кортежи, входящие в отношение-первый операнд, такие, что ни один из них не входит в отношение, которое является вторым операндом.

При выполнении декартова произведения (TIMES) двух отношений, пересечение заголовков которых пусто, производится отношение, кортежи которого производятся путем объединения кортежей первого и второго операндов.

Результатом ограничения (WHERE) отношения по некоторому условию является отношение, включающее кортежи отношения-операнда, удовлетворяющее этому условию.

При выполнении проекции (PROJECT) отношения на заданное подмножество множества его атрибутов производится отношение, кортежи которого являются соответствующими подмножествами кортежей отношения-операнда.

При соединении (JOIN) двух отношений по некоторому условию образуется результирующее отношение, кортежи которого производятся путем объединения кортежей первого и второго отношений и удовлетворяют этому условию.

У операции реляционного деления (DIVIDE BY) два операнда – бинарное и унарное отношения. Результирующее отношение состоит из унарных кортежей, включающих значения первого атрибута кортежей первого операнда таких, что множество значений второго атрибута (при фиксированном значении первого атрибута) включает множество значений второго операнда.

Операция переименования (RENAME) производит отношение, тело которого совпадает с телом операнда, но имена атрибутов изменены.

Операция присваивания (:=) позволяет сохранить результат вычисления реляционного выражения в существующем отношении БД.

Поскольку результатом любой реляционной операции (кроме операции присваивания, которая не вырабатывает значения) является некое отношение, можно образовывать реляционные выражения, в которых вместо отношения-операнда некоторой реляционной операции находится вложенное реляционное выражение. В построении реляционного выражения могут участвовать все реляционные операции, кроме операции присваивания. Вычислительная интерпретация реляционного выражения диктуется установленными приоритетами операций:

RENAME WHERE = PROJECT TIMES = JOIN = INTERSECT = DIVIDE BY UNION = MINUS

В другой форме приоритеты операций показаны на рис. 1. Вычисление выражения производится слева направо с учетом приоритетов операций и скобок.

Операция	Приоритет
RENAME	4
WHERE	3
PROJECT	3
TIMES	2
JOIN	2
INTERSECT	2
DIVIDE BY	2
UNION	1
MINUS	1

Рис. 1. Таблица приоритетов операций традиционной реляционной алгебры

Замкнутость реляционной алгебры и операция переименования

Каждое значение-отношение характеризуется заголовком (или схемой) и телом (или множеством кортежей). Поэтому, если нам действительно нужна алгебра, операции которой замкнуты относительно понятия отношения, то каждая операция должна производить отношение в полном смысле, т. е. оно должно обладать и телом, и заголовком. Только в этом случае можно будет строить вложенные выражения.

Заголовок отношения представляет собой множество пар <имя-атрибута, имя-домена>. Если посмотреть на общий обзор реляционных операций, приведенный в предыдущем подразделе, то видно, что домены атрибутов результирующего отношения однозначно определяются доменами отношений-операндов. Однако с именами атрибутов результата не всегда все так просто.

Например, представим себе, что у отношений-операндов операции декартова произведения имеются одноименные атрибуты с одинаковыми доменами. Каким был бы заголовок результирующего отношения? Поскольку это множество, в нем не должны содержаться одинаковые элементы. Но и потерять атрибут в результате недопустимо. А это значит, что в таком случае вообще невозможно корректно выполнить операцию декартова произведения.

Аналогичные проблемы могут возникать и в случаях других двуместных операций. Для разрешения проблем в число операций реляционной алгебры вводится операция переименования. Ее следует применять в том случае, когда возникает конфликт именования атрибутов в отношениях-

операндах одной реляционной операции. Тогда к одному из операндов сначала применяется операция переименования, а затем основная операция выполняется уже без всяких проблем. Более строго мы определим операцию переименования в следующей лекции, а пока лишь заметим, что результатом этой операции является отношение, совпадающее во всем с отношением-операндом, кроме того, что имя указанного атрибута изменено на заданное имя.

В дальнейшем изложении мы будем предполагать применение операции переименования во всех конфликтных ситуациях. Заметим, кстати, что невозможность применения некоторых операций к произвольным парам значений отношений без предварительного переименования атрибутов отношений операндов означает, что «алгебра» Кодда не является алгеброй отношений в математическом смысле. Описываемая в следующей главе Алгебра А такими недостатками не обладает: результотатом применения любой операции к любым отношениям является некоторое отношение.

Особенности теоретико-множественных операций реляционной алгебры

Хотя в основе теоретико-множественной части реляционной алгебры Кодда лежит классическая теория множеств, соответствующие операции реляционной алгебры обладают некоторыми особенностями.

Операции объединения, пересечения, взятия разности. Совместимость по объединению

Начнем с операции объединения отношений (все, что будет сказано по поводу объединения, верно и для операций пересечения и взятия разности отношений). Смысл операции объединения в реляционной алгебре в целом остается теоретико-множественным. В теории множеств:

-результатом объединения двух множеств A{a} и B{b} является такое множество C{c}, что для каждого c либо существует такой элемент a, принадлежащий множеству A, что c=a, либо существует такой элемент b, принадлежащий множеству B, что c=b;

-пересечением множеств A и B является такое множество C{c}, что для любого c существуют такие элементы a, принадлежащий множеству A, и b, принадлежащий множеству B, что c=a=b;

-разностью множеств A и B является такое множество C{c}, что для любого c существует такой элемент a, принадлежащий множеству A, что c=a, и не существует такой элемент b, принадлежащий B, что c=b.

Но если в теории множеств операция объединения осмысленна для любых двух множеств-операндов, то в случае реляционной алгебры результатом операции объединения должно являться отношение. Если в реляционной алгебре допустить возможность теоретико-множественного объединения двух произвольных отношений (с разными заголовками), то, конечно, результатом операции будет множество, но множество разнотипных кортежей, т. е. не отношение. Если исходить из требования замкнутости реляционной алгебры относительно понятия отношения, то такая операция объединения является бессмысленной.

Эти соображения подводят к понятию совместимости отношений по объединению: два отношения совместимы по объединению в том и только в том случае, когда обладают одинаковыми заголовками. В развернутой форме это означает, что в заголовках обоих отношений содержится один и тот же набор имен атрибутов, и одноименные атрибуты определены на одном и том же домене (эта развернутая формулировка, вообще говоря, является излишней, но она пригодится нам в следующем абзаце).

Если два отношения совместимы по объединению, то при обычном выполнении над ними операций объединения, пересечения и взятия разности результатом операции является отношение с корректно определенным заголовком, совпадающим с заголовком каждого из отношений-операндов. Напомним, что если два отношения «почти» совместимы по объединению, т. е. совместимы во всем, кроме имен атрибутов, то до выполнения операции типа объединения эти отношения можно сделать полностью совместимыми по объединению путем применения операции переименования.

Для иллюстрации операций объединения, пересечения и взятия разности предположим, что в базе данных имеются два отношения СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 и СЛУЖАЩИЕ_В_ПРОЕКТЕ_2

с одинаковыми схемами {СЛУ_НОМЕР, СЛУ_ИМЯ, СЛУ_ЗАРП, СЛУ_ОТД_НОМЕР} (имена доменов опущены по причине очевидности). Каждое из отношений содержит данные о служащих, участвующих в соответствующем проекте. На рис. 2 показано примерное наполнение каждого из двух отношений (некоторые служащие участвуют в обоих проектах).

СЛУЖАЩИЕ_В_ПРОЕКТЕ_1

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2934	Иванов	22000.00	310
2935	Петров	30000.00	310
2936	Сидоров	18000.00	313
2937	Федоров	20000.00	310
2938	Иванова	22000.00	315

СЛУЖАЩИЕ_В_ПРОЕКТЕ_2

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2934	Иванов	22000.00	310
2935	Петров	30000.00	310
2939	Сидоренко	18000.00	313
2940	Федоренко	20000.00	310
2941	Иваненко	22000.00	315

Рис. 2. Примерное наполнение отношений СЛУЖАЩИЕ _В_ПРОЕКТЕ_1 и СЛУЖАЩИЕ _В_ПРОЕКТЕ_2

Тогда выполнение операции СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 UNION СЛУЖАЩИЕ_В_ПРОЕКТЕ_2 позволит получить информацию обо всех служащих, участвующих в обоих проектах. Выполнение операции СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 INTERSECT СЛУЖАЩИЕ_В_ПРОЕКТЕ_2 позволит получить данные о служащих, которые одновременно участвуют в двух проектах. Наконец, операция СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 MINUS СЛУЖАЩИЕ_В_ПРОЕКТЕ_2 выработает отношение, содержащее кортежи служащих, которые участвуют только в первом проекте. Результаты этих операций показаны на рис. 3.

СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 UNION **СЛУЖАЩИЕ_В_ПРОЕКТЕ_2**

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2934	Иванов	22000.00	310
2935	Петров	30000.00	310
2939	Сидоренко	18000.00	313
2940	Федоренко	20000.00	310
2941	Иваненко	22000.00	315
2936	Сидоров	18000.00	313
2937	Федоров	20000.00	310
2938	Иванова	22000.00	315

СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 INTERSECT **СЛУЖАЩИЕ_В_ПРОЕКТЕ_2**

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2934	Иванов	22000.00	310
2935	Петров	30000.00	310

СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 MINUS **СЛУЖАЩИЕ_В_ПРОЕКТЕ_2**

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2936	Сидоров	18000.00	313
2937	Федоров	20000.00	310
2938	Иванова	22000.00	315

Рис. 3. Результаты выполнения операций UNION, INTERSECT и MINUS

Заметим, что включение в состав операций реляционной алгебры трех операций объединения, пересечения и взятия разности является, очевидно, избыточным, поскольку, например, операция пересечения выражается через операцию взятия разности14). Тем не менее Кодд в свое время решил включить все три операции, исходя из интуитивных потребностей далекого от математики потенциального пользователя системы реляционных БД.

Операция расширенного декартова произведения и совместимость отношений относительно этой операции

Другие проблемы связаны с операцией взятия декартова произведения двух отношений. В теории множеств декартово произведение может быть получено для любых двух множеств, и элементами результирующего множества являются пары, составленные из элементов первого и второго множеств. Если говорить более точно, декартовым произведением множеств A{a} и B{b} является такое множество пар C{<c1, 2>}, что для каждого элемента <c1, c2> множества C существуют такой элемент a множества A, что c1=a, и такой элемент b множества B, что c2=b.

Поскольку отношения являются множествами, для любых двух отношений возможно получение прямого произведения. Но результат не будет отношением! Элементами результата будут не кортежи, а пары кортежей.

Поэтому в реляционной алгебре используется специализированная форма операции взятия декартова произведения – расширенное декартово произведение отношений. При взятии расширенного декартова произведения двух отношений элементом результирующего отношения является кортеж, который представляет собой объединение одного кортежа первого отношения и одного кортежа второго отношения.

Приведем более точное определение операции расширенного декартова произведения. Пусть имеются два отношения R1{a1, a2, …, an} и R2{b1, b2, …, bm}. Тогда результатом операции R1 TIMES R2 является отношение R{a1, a2, …, an, b1, b2, …, bm}, тело которого является множеством кортежей вида {ra1, ra2, …, ran, rb1, rb2, …, rbm} таких, что {ra1, ra2, …, ran} входит в тело R1, а {rb1, rb2, …, rbm} входит в тело R2.

Но теперь возникает вторая проблема – как получить корректно сформированный заголовок отношения-результата? Поскольку схема результирующего отношения является объединением схем отношений-операндов, то очевидной проблемой может быть именование атрибутов результирующего отношения, если отношения-операнды обладают одноименными атрибутами.

Эти соображения приводят к введению понятия совместимости по взятию расширенного декартова произведения. Два отношения совместимы по взятию расширенного декартова произведения в том и только в том случае, если пересечение множеств имен атрибутов, взятых из их схем отношений, пусто. Любые два отношения всегда могут стать совместимыми по взятию декартова произведения, если применить операцию переименования к одному из этих отношений.

Для наглядности предположим, что в придачу к введенным ранее отношениям СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 и СЛУЖАЩИЕ_В_ПРОЕКТЕ_2 в базе данных содержится еще и отношение ПРОЕКТЫ со схемой {ПРОЕКТ_НАЗВ, ПРОЕКТ_РУК} (имена доменов снова опущены) и телом, показанным на рис. 4. На этом же рисунке показан результат операции СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 TIMES ПРОЕКТЫ.

ПРОЕКТЫ

ПРОЕКТ_НАЗВ	ПРОЕКТ_РУК
ПРОЕКТ 1	Иванов
ПРОЕКТ 2	Иваненко

СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 TIMES **ПРОЕКТЫ**

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР	ПРОЕКТ_НАЗВ	ПРОЕКТ_РУК
2934	Иванов	22000.00	310	ПРОЕКТ 1	Иванов
2935	Петров	30000.00	310	ПРОЕКТ 1	Иванов
2936	Сидоров	18000.00	313	ПРОЕКТ 1	Иванов
2937	Федоров	20000.00	310	ПРОЕКТ 1	Иванов
2938	Иванова	22000.00	315	ПРОЕКТ 1	Иванов
2934	Иванов	22000.00	310	ПРОЕКТ 2	Иваненко
2935	Петров	30000.00	310	ПРОЕКТ 2	Иваненко
2936	Сидоров	18000.00	313	ПРОЕКТ 2	Иваненко
2937	Федоров	20000.00	310	ПРОЕКТ 2	Иваненко
2938	Иванова	22000.00	315	ПРОЕКТ 2	Иваненко

Рис. 4. Отношение ПРОЕКТЫ и результат операции СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 TIMES ПРОЕКТЫ

Следует заметить, что операция взятия декартова произведения не является слишком осмысленной на практике. Во-первых, мощность тела ее результата очень велика даже при допустимых мощностях операндов, а во-вторых, результат операции не более информативен, чем взятые в совокупности операнды. Как будет показано далее, основной смысл включения операции расширенного декартова произведения в состав реляционной алгебры Кодда состоит в том, что на ее основе определяется действительно полезная операция соединения.

По поводу теоретико-множественных операций реляционной алгебры следует еще заметить, что все четыре операции являются ассоциативными. Т. е. если обозначить через OP любую из четырех операций, то (A OP B) OP C = A OP (B OP C), и, следовательно, без внесения двусмысленности можно писать A OP B OP C (A, B и C – отношения, обладающие свойствами, необходимыми для корректного выполнения соответствующей операции). Все операции, кроме взятия разности, являются коммутативными, т. е. A OP B = B OP A.

Специальные реляционные операции

Операция ограничения

Операция ограничения требует наличия двух операндов: ограничиваемого отношения и простого условия ограничения. Простое условие ограничения может иметь либо вид (a comp-op b), где a и b - имена атрибутов ограничиваемого отношения, для которых осмысленна операция сравнения comp-op, либо вид (a comp-op const), где a - имя атрибута ограничиваемого отношения, a const - литерально заданная константа.

В результате выполнения операции ограничения производится отношение, заголовок которого совпадает с заголовком отношения-операнда, а в тело входят те кортежи отношения-операнда, для которых значением условия ограничения является true.

Пусть UNION обозначает операцию объединения, INTERSECT - операцию пересечения, а MINUS - операцию взятия разности. Для обозначения операции ограничения будем использовать конструкцию A WHERE comp, где A - ограничиваемое отношение, а comp - простое условие сравнения. Пусть comp1 и comp2 - два простых условия ограничения. Тогда по определению:

A WHERE comp1 AND comp2 обозначает то же самое, что и (A WHERE comp1) INTERSECT (A WHERE comp2)

A WHERE comp1 OR comp2 обозначает то же самое, что и (A WHERE comp1) UNION (A WHERE comp2)

A WHERE NOT comp1 обозначает то же самое, что и A MINUS (A WHERE comp1)

С использованием этих определений можно использовать операции ограничения, в которых условием ограничения является произвольное булевское выражение, составленное из простых условий с использованием логических связок AND, OR, NOT и скобок.

На интуитивном уровне операцию ограничения лучше всего представлять как взятие некоторой "горизонтальной" вырезки из отношения-операнда.

Операция взятия проекции

Операция взятия проекции также требует наличия двух операндов - проецируемого отношения A и списка имен атрибутов, входящих в заголовок отношения A.

Результатом проекции отношения A по списку атрибутов a1, a2, ..., an является отношение, с заголовком, определяемым множеством атрибутов a1, a2, ..., an, и с телом, состоящим из кортежей вида <a1:v1, a2:v2, ..., an:vn> таких, что в отношении A имеется кортеж, атрибут a1 которого имеет значение v1, атрибут a2 имеет значение v2, ..., атрибут an имеет значение vn. Тем самым, при выполнении операции проекции выделяется "вертикальная" вырезка отношения-операнда с естественным уничтожением потенциально возникающих кортежей-дубликатов.

Операция соединения отношений

Общая операция соединения (называемая также соединением по условию) требует наличия двух операндов - соединяемых отношений и третьего операнда - простого условия. Пусть соединяются отношения A и B. Как и в случае операции ограничения, условие соединения comp имеет вид либо (a comp-op b), либо (a comp-op const), где a и b - имена атрибутов отношений A и B, const - литерально заданная константа, а comp-op - допустимая в данном контексте операция сравнения.

Тогда по определению результатом операции сравнения является отношение, получаемое путем выполнения операции ограничения по условию comp прямого произведения отношений A и B.

Если внимательно осмыслить это определение, то станет ясно, что в общем случае применение условия соединения существенно уменьшит мощность результата промежуточного прямого произведения отношений-операндов только в том случае, когда условие соединения имеет вид (a comp-op b), где a и b - имена атрибутов разных отношений-операндов. Поэтому на практике обычно считают реальными операциями соединения именно те операции, которые основываются на условии соединения приведенного вида.

Хотя операция соединение в нашей интерпретации не является примитивной (поскольку она определяется с использованием прямого произведения и проекции), в силу особой практической важности она включается в базовый набор операций реляционной алгебры. Заметим также, что в практических реализациях соединение обычно не выполняется именно как ограничение прямого произведения. Имеются более эффективные алгоритмы, гарантирующие получение такого же результата.

Имеется важный частный случай соединения - эквисоединение и простое, но важное расширение операции эквисоединения - естественное соединение. Операция соединения называется операцией эквисоединения, если условие соединения имеет вид (a = b), где a и b - атрибуты разных операндов соединения. Этот случай важен потому, что (a) он часто встречается на практике, и (b) для него существуют эффективные алгоритмы реализации.

Операция естественного соединения применяется к паре отношений A и B, обладающих (возможно составным) общим атрибутом c (т.е. атрибутом с одним и тем же именем и определенным на одном и том же домене). Пусть ab обозначает объединение заголовков отношений A и B. Тогда естественное соединение A и B - это спроектированный на ab результат эквисоединения A и B по A/c и BBC. Если вспомнить введенное нами в конце предыдущей главы определение внешнего ключа отношения, то должно стать понятно, что основной смысл операции естественного соединения - возможность

восстановления сложной сущности, декомпозированной по причине требования первой нормальной формы. Операция естественного соединения не включается прямо в состав набора операций реляционной алгебры, но она имеет очень важное практическое значение.

Операция деления отношений

Эта операция наименее очевидна из всех операций реляционной алгебры и поэтому нуждается в более подробном объяснении. Пусть заданы два отношения - A с заголовком {a1, a2, ..., an, b1, b2, ..., bm} и B с заголовком {b1, b2, ..., bm}. Будем считать, что атрибут bi отношения A и атрибут bi отношения B не только обладают одним и тем же именем, но и определены на одном и том же домене. Назовем множество атрибутов {aj} составным атрибутом a, а множество атрибутов {bj} - составным атрибутом b. После этого будем говорить о реляционном делении бинарного отношения A(a,b) на унарное отношение B(b).

Результатом деления A на B является унарное отношение C(a), состоящее из кортежей v таких, что в отношении A имеются кортежи <v, w> такие, что множество значений {w} включает множество значений атрибута b в отношении B.

Предположим, что в базе данных сотрудников поддерживаются два отношения: СОТРУДНИКИ (ИМЯ, ОТД_НОМЕР) и ИМЕНА (ИМЯ), причем унарное отношение ИМЕНА содержит все фамилии, которыми обладают сотрудники организации. Тогда после выполнения операции реляционного деления отношения СОТРУДНИКИ на отношение ИМЕНА будет получено унарное отношение, содержащее номера отделов, сотрудники которых обладают всеми возможными в этой организации именами.

3. Обзор реляционной алгебры А Дейта и Дарвена.

Базисом предложенной Крисом Дейтом и Хью Дарвеном Алгебры А являются операции реляционного отрицания (дополнения), реляционной конъюнкции (или дизъюнкции) и проекции (удаления атрибута). Реляционные аналоги логических операций определяются в терминах отношений на основе обычных теоретико-множественных операций и позволяют выражать напрямую операции пересечения, декартова произведения, естественного соединения, объединения отношений и т. д. Путем комбинирования базовых операций выражаются операции переименования атрибутов, соединения общего вида, взятия разности отношений. Алгебра А позволяет лучше осознать логические основы реляционной модели, хотя, безусловно, является в меньшей степени ориентированной на практическое применение, чем алгебра Кодда16). Даже сами авторы Алгебры А, Дейт и Дарвен, в своем учебном языке Tutorial D используют не Алгебру А напрямую, а некоторое ее надмножество, в большей степени напоминающее алгебру Кодда.

Базовые операции Алгебры А

Пусть r – отношение, A – имя атрибута отношения r, T – имя соответствующего типа (т. е. типа или домена атрибута A), v – значение типа T. Тогда:

-заголовком Hr отношения r называется множество атрибутов, т.е. упорядоченных пар вида <A, T>. По определению никакие два атрибута в этом множестве не могут содержать одно и то же имя атрибута A;

-кортеж tr, соответствующий заголовку Hr, – это множество упорядоченных триплетов вида <A, T, v>, по одному такому триплету для каждого атрибута в Hr;

-тело Br отношения r – это множество кортежей tr. Заметим, что (в общем случае) могут существовать такие кортежи tr, которые соответствуют Hr, но не входят в Br.

Заметим, что заголовок – это множество (упорядоченных пар вида <A, T>), тело – это множество (кортежей tr), и кортеж – это множество (упорядоченных триплетов вида <A, T, v>). Элемент заголовка – это атрибут (т. е. упорядоченная пара вида <A,T>); элемент тела – это кортеж; элемент кортежа – это упорядоченный триплет вида <A, T, v>. Любое подмножество заголовка – это заголовок, любое подмножество тела – это тело, и любое подмножество кортежа – это кортеж.

Определим несколько основных операций (как будет показано далее, некоторые из них избыточны). Каждое из последующих определений состоит из: формальной спецификации ограничений (если они имеются), применимых к операндам соответствующей

операции; формальной спецификации заголовка результата этой операции; формальной спецификации тела этого результата и неформального обсуждения формальных спецификаций.

Во всех формальных спецификациях exists обозначает квантор существования; exists tr означает «существует такой tr, что». Символ «» означает принадлежность одного множества другому; trBr означает, что элемент tr принадлежит множеству Br. Выражение trBr означает, что элемент tr не принадлежит множеству Br. Операции minus и union являются традиционными теоретико-множественными операциями взятия разности и объединения множеств.

Поскольку некоторые базовые операции Алгебры А имеют названия обычных логических операций, чтобы избежать путаницы, имена реляционных операций берутся в угловые скобки: <NOT>, <AND>, <OR> и т. д. В исходный базовый набор операций входят операции реляционного дополнения <NOT>, удаления атрибута <REMOVE>, переименования атрибута <RENAME>, реляционной конъюнкции <AND> и реляционной дизъюнкции <OR>.

Операция реляционного дополнения

Пусть s обозначает результат операции <NOT> r. Тогда:

Hs = Hr (заголовок результата совпадает с заголовком операнда);

Bs = {ts : exists tr (tr Br and ts = tr) } (в тело результата входят все кортежи, соответствующие заголовку и не входящие в тело операнда).

Операция <NOT> производит дополнение s заданного отношения r. Заголовком s является заголовок r. Тело s включает все кортежи, соответствующие этому заголовку и не входящие в тело r.

Во-первых, термин «дополнение» полностью соответствует сути операции <NOT>: тело результата операции <NOT> r является дополнением Br до полного множества кортежей, соответствующих Hr. Во-вторых, это не противоречит природе булевской операции NOT: у булевского типа имеются всего два значения – true и false, и NOT true = false, a NOT false = true.

Чтобы привести пример использования операции <NOT>, предположим, что в состав домена ДОПУСТИМЫЕ_НОМЕРА_ПРОЕКТОВ, на котором определен атрибут ПРО_НОМ отношения НОМЕРА_ПРОЕКТОВ с рис. 5 слева, входит всего пять значений {1, 2, 3, 4, 5}. Тогда результат операции <NOT> НОМЕРА_ПРОЕКТОВ будет таким, как показано на рис. 5 справа.

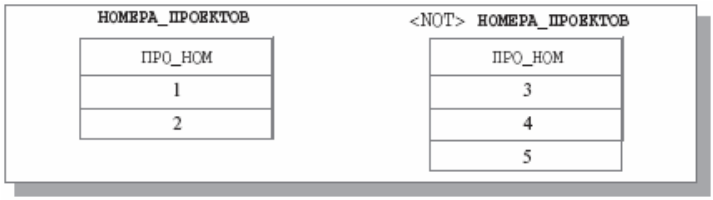


Рис. 5. Результат операции <NOT> НОМЕРА_ПРОЕКТОВ

Операция удаления атрибута

Пусть s обозначает результат операции r <REMOVE> A. Для обеспечения возможности выполнения операции требуется, чтобы существовал некоторый тип (или домен) T такой, что <A, T> Hr (т. е. в состав заголовка отношения r должен входить атрибут A). Тогда:

-Hs = Hr minus {<A, T>}, т. е. заголовок результата получается из заголовка операнда изъятием атрибута A;

-Bs = {ts : exists tr exists v (tr Br and v T and <A,T,v> tr and ts = tr minus {<A,T,v>})}, т. е. в тело результата входят все кортежи операнда, из которых удалено значение атрибута A.

Операция <REMOVE> производит отношение s, формируемое путем удаления указанного атрибута A из заданного отношения r. Операция эквивалентна взятию проекции r

на все атрибуты, кроме A. Заголовок s получается теоретико-множественным вычитанием из заголовка r множества из одного элемента {<A, T>}. Тело s состоит из таких кортежей, которые соответствуют заголовку s, причем каждый из них является подмножеством некоторого кортежа тела отношения r.

Примером операции REMOVE (конечно же, очень похожим на пример использования операции PROJECT из предыдущей лекции) является СЛУЖАЩИЕ REMOVE ПРО_НОМ (получить данные о служащих, участвующих в проектах). Результат этой операции над отношением СЛУЖАЩИЕ, тело которого приведено в верхней части рис. 6, показан на рис. 5.2 внизу.

СЛУЖАЩИЕ

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2934	Иванов	22400.00	1
2935	Петров	29600.00	1
2936	Сидоров	18000.00	1
2937	Федоров	20000.00	1
2938	Иванова	22000.00	1
2934	Иванов	22400.00	2
2935	Петров	29600.00	2
2939	Сидоренко	18000.00	2
2940	Федоренко	20000.00	2
2941	Иваненко	22000.00	2

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП
2934	Иванов	22400.00
2935	Петров	29600.00
2936	Сидоров	18000.00
2937	Федоров	20000.00
2938	Иванова	22000.00
2939	Сидоренко	18000.00
2940	Федоренко	20000.00
2941	Иваненко	22000.00

Рис. 6. Результат операции СЛУЖАЩИЕ REMOVE ПРО_НОМ

Операция переименования

Пусть s обозначает результат операции r <RENAME> (A, B). Для обеспечения возможности выполнения операции требуется, чтобы существовал некоторый тип T, такой, что <A, T> Hr, и чтобы не существовал такой тип T, что <B, T> Hr. (Другими словами, в схеме отношения r должен присутствовать атрибут A и не должен присутствовать атрибут B.) Тогда:

-Hs = (Hr minus {<A, T>}) union {<B, T>}, т. е. в схеме результата B заменяет A;

-Bs = {ts : exists tr exists v (tr Br and v T and <A, T, v> tr and ts = (tr minus {<A, T, v>}) union {<B, T, v>})}, т. е. в кортежах тела результата имя значений атрибута A меняется на B.

Операция <RENAME> производит отношение s, которое отличается от заданного отношения r только именем одного его атрибута, которое изменяется с A на B. Заголовок s такой же, как заголовок r, за исключением того, что пара <B, T> заменяет пару <A, T>. Тело s включает все кортежи тела r, но в каждом из этих кортежей триплет <B, T, v> заменяет триплет <A, T, v>.

Операция реляционной конъюнкции

Пусть s обозначает результат операции r1 <AND> r2. Для обеспечения возможности выполнения операции требуется, чтобы если <A, T1>Hr1 и <A, T2>Hr2, то T1=T2. (Другими словами, если в двух отношениях-операндах имеются одноименные атрибуты, то они должны быть определены на одном и том же типе (домене).) Тогда:

-Hs = Hr1 union Hr2, т. е. заголовок результата получается путем объединения заголовков отношений-операндов, как в операциях TIMES и JOIN из предыдущей лекции;

-Bs = { ts : exists tr1 exists tr2 ((tr1Br1 and tr2Br2) and ts = tr1 union tr2)}; обратите внимание на то, что кортеж результата определяется как объединение кортежей операндов; поэтому:

-если схемы отношений-операндов имеют непустое пересечение, то операция <AND> работает как естественное соединение;

-если пересечение схем операндов пусто, то <AND> работает как расширенное декартово произведение;

-если схемы отношений полностью совпадают, то результатом операции является пересечение двух отношений-операндов.

Операция <AND> является реляционной конъюнкцией, в некоторых случаях выдающей в результате отношение rs, ранее называвшееся естественным соединением двух заданных отношений r1 и r2. Заголовок rs является объединением заголовков r1 и r2. Тело s включает каждый кортеж, соответствующий заголовку s и являющийся надмножеством некоторого кортежа из тела r1 и некоторого кортежа из тела r2.

Для иллюстрации воспользуемся примерными отношениями, показанными на рис. 7.

СЛУЖАЩИЕ_В_ПРОЕКТЕ_1

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2934	Иванов	22000.00	310
2935	Петров	30000.00	310
2936	Сидоров	18000.00	313
2937	Федоров	20000.00	310
2938	Иванова	22000.00	315

СЛУЖАЩИЕ_В_ПРОЕКТЕ_2

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2934	Иванов	22000.00	310
2935	Петров	30000.00	310
2939	Сидоренко	18000.00	313
2940	Федоренко	20000.00	310
2941	Иваненко	22000.00	315

ПРОЕКТЫ

ПРО_НОМ	ПРОЕКТ_РУК
1	Иванов
2	Иваненко

СЛУЖАЩИЕ

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2934	Иванов	22400.00	1
2935	Петров	29600.00	1
2936	Сидоров	18000.00	1
2937	Федоров	20000.00	1
2938	Иванова	22000.00	1
2934	Иванов	22400.00	2
2935	Петров	29600.00	2
2939	Сидоренко	18000.00	2
2940	Федоренко	20000.00	2
2941	Иваненко	22000.00	2

Рис. 7. Примерные отношения для иллюстрации операции <AND>

На рис. 8(a) у отношений СЛУЖАЩИЕ и ПРОЕКТЫ имеется общий атрибут ПРО_НОМ. Поэтому операция <AND> работает как операция естественного соединения. На рис. 8(b) пересечение заголовков отношений СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 и ПРОЕКТЫ пусто, и поэтому в результате реляционной конъюнкции производится расширенное декартово произведение этих отношений. Наконец, на рис. 8(c) схемы отношений СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 и СЛУЖАЩИЕ_В_ПРОЕКТЕ_2 совпадают, и телом операции <AND> является пересечение тел отношений-операндов.

(a) Результат операции **СЛУЖАЩИЕ** <AND> **ПРОЕКТЫ**

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ	ПРОЕКТ_РУК
2934	Иванов	22400.00	1	Иванов
2935	Петров	29600.00	1	Иванов
2936	Сидоров	18000.00	1	Иванов
2937	Федоров	20000.00	1	Иванов
2938	Иванова	22000.00	1	Иванов
2934	Иванов	22400.00	2	Иваненко
2935	Петров	29600.00	2	Иваненко
2939	Сидоренко	18000.00	2	Иваненко
2940	Федоренко	20000.00	2	Иваненко
2941	Иваненко	22000.00	2	Иваненко

(b) Результат операции **СЛУЖАЩИЕ_В_ПРОЕКТЕ_1** <AND> **ПРОЕКТЫ**

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР	ПРО_НОМ	ПРОЕКТ_РУК
2934	Иванов	22000.00	310	1	Иванов
2935	Петров	30000.00	310	1	Иванов
2936	Сидоров	18000.00	313	1	Иванов
2937	Федоров	20000.00	310	1	Иванов
2938	Иванова	22000.00	315	1	Иванов
2934	Иванов	22000.00	310	2	Иваненко
2935	Петров	30000.00	310	2	Иваненко
2936	Сидоров	18000.00	313	2	Иваненко
2937	Федоров	20000.00	310	2	Иваненко
2938	Иванова	22000.00	315	2	Иваненко

(c) Результат операции **СЛУЖАЩИЕ_В_ПРОЕКТЕ_1** <AND> **СЛУЖАЩИЕ_В_ПРОЕКТЕ_2**

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2934	Иванов	22000.00	310
2935	Петров	30000.00	310

Рис. 8. Иллюстрации операции реляционной конъюнкции

Операция реляционной дизъюнкции

Пусть s обозначает результат операции r1 <OR> r2. Для обеспечения возможности выполнения операции требуется, чтобы если <A, T1>Hr1 и <A, T2>Hr2, то должно быть T1 = T2 (одноименные атрибуты должны быть определены на одном и том же типе). Тогда:

-Hs = Hr1 union Hr2 (из схемы результата удаляются атрибуты-дубликаты);

-Bs = { ts : exists tr1 exists tr2 ((tr1Br1 or tr2Br2) and ts = tr1 union tr2)}; очевидно, что при этом:

-если у операндов нет общих атрибутов, то в тело результирующего отношения входят все такие кортежи ts, которые являются объединением кортежей tr1 и tr2, соответствующих заголовкам отношений-операндов, и хотя бы один из этих кортежей принадлежит телу одного из операндов;

-если у операндов имеются общие атрибуты, то в тело результирующего отношения входят все такие кортежи ts, которые являются объединением кортежей tr1 и tr2, соответствующих заголовкам отношений-операндов, если хотя бы один из этих кортежей принадлежит телу одного из операндов, и значения общих атрибутов tr1 и tr2 совпадают;

-если же схемы отношений-операндов совпадают, то тело отношения-результата является объединением тел операндов.

Операция <OR> является реляционной дизъюнкцией и обобщением того, что ранее называлось объединением. Заголовок s есть объединение заголовков r1 и r2. Тело s состоит из всех кортежей, соответствующих заголовку s и являющихся надмножеством либо некоторого кортежа из тела r1, либо некоторого кортежа из тела r2.

Предположим, у нас имеются отношения ПРОЕКТЫ_1 {ПРОЕКТ_НАЗВ, ПРОЕКТ_РУК} и НОМЕРА_ПРОЕКТОВ {ПРО_НОМ} (рис. 5.5). Предположим также, что

домен атрибута ПРОЕКТ_НАЗВ включает значения ПРОЕКТ_1, ПРОЕКТ_2, ПРОЕКТ_3, домен атрибута ПРОЕКТ_РУК ограничен значениями Иванов, Иваненко, а доменом атрибута ПРО_НОМ является множество {1, 2, 3}. Результат операции ПРОЕКТЫ <OR> НОМЕРА_ПРОЕКТОВ показан на рис. 9.

Как показано на рис. 9, операция <OR> при наличии операндов с несовпадающими схемами производит результат, гораздо более мощный, чем результат операции взятия расширенного декартова произведения из лекции 4, и еще менее осмысленный с практической точки зрения.

Для иллюстрации операции <OR> над операндами, схемы которых имеют непустое пересечение, воспользуемся отношением ПРОЕКТЫ_2 {ПРО_НОМ, ПРОЕКТ_РУК} (рис. 10) и унарным отношением НОМЕРА_ПРОЕКТОВ, схема и тело которого показаны на рис. 9. Будем предполагать, что множества значений доменов атрибутов такие же, как в предыдущем примере. Результат операции ПРОЕКТЫ_2 <OR> НОМЕРА_ПРОЕКТОВ показан на рис. 10.

Как уже отмечалось, при совпадении схем отношений-операндов результатом выполнения над ними операции <OR> является объединение отношений. Это непосредственно следует из спецификации операции. Если этот факт кажется неочевидным, еще раз внимательно посмотрите на спецификацию. Иллюстрирующий пример мы приводить не будем.

ПРОЕКТЫ_1

ПРОЕКТ_НАЗВ	ПРОЕКТ_РУК
ПРОЕКТ 1	Иванов
ПРОЕКТ 2	Иваненко

НОМЕРА_ПРОЕКТОВ

ПРО_НОМ
1
2

Результат операции **ПРОЕКТЫ** <OR> **НОМЕРА_ПРОЕКТОВ**

ПРОЕКТ_НАЗВ	ПРОЕКТ_РУК	ПРО_НОМ
ПРОЕКТ 1	Иванов	1
ПРОЕКТ 2	Иванов	1
ПРОЕКТ 3	Иванов	1
ПРОЕКТ 1	Иваненко	1
ПРОЕКТ 2	Иваненко	1
ПРОЕКТ 3	Иваненко	1
ПРОЕКТ 1	Иванов	2
ПРОЕКТ 2	Иванов	2
ПРОЕКТ 3	Иванов	2
ПРОЕКТ 1	Иваненко	2
ПРОЕКТ 2	Иваненко	2
ПРОЕКТ 3	Иваненко	2
ПРОЕКТ 1	Иванов	3
ПРОЕКТ 2	Иваненко	3

Рис. 9. Результат операции <OR> над операндами без общих атрибутов

ПРОЕКТЫ_2

ПРО_НОМ	ПРОЕКТ_РУК
1	Иванов
2	Иваненко

ПРОЕКТЫ_2 <OR> **НОМЕРА_ПРОЕКТОВ**

ПРО_НОМ	ПРОЕКТ_РУК
1	Иванов
2	Иваненко
2	Иванов
1	Иваненко

Рис. 10. Результат операции <OR> над операндами, схемы которых частично пересекаются

1. 6 Лекция №6 (2 часа).

Тема: «Основы проектирования баз данных»

1.6.1 Вопросы лекции:

1. Основные понятия баз данных

2. Основы проектирования баз данных

3. Реляционный подход к организации данных в базах данных

1.6.2 Краткое содержание вопросов:

1. Основные понятия баз данных

Электронные картотеки на материальных носителях, в которых данные структурированы таким образом, чтобы их могли использовать различные пользователи и программы, носят название Баз данных (БД). Комплекс программных и языковых средств для создания баз данных, поддержания их в актуальном состоянии и организации поиска в них необходимой информации называется Системой управления базами данных (СУБД).

По технологии обработки данных БД подразделяются на централизованные и распределенные. Централизованная БД хранится в памяти одной вычислительной системы. Если эта вычислительная система является компонентом сети ЭВМ, возможен распределенный доступ (сетевой) к такой базе. Такой способ использования БД часто применяют в локальных сетях ПК. Распределенная БД состоит из нескольких, иногда пересекающихся или дублирующих друг друга частей, которые хранятся в памяти различных ЭВМ вычислительной сети. Работа с такой базой осуществляется с помощью Системы управления распределенной БД (СУРБД).

По способу доступа к данным БД разделяются на БД с локальным доступом и БД с сетевым (удаленным) доступом. Системы централизованных БД с сетевым доступом предполагают две основные архитектуры: Файл-сервер, Клиент-сервер.

Архитектура Файл-сервер предполагает выделение одной из машин сети в качестве центральной (сервер файлов), на которой хранится совместно используемая централизованная БД. Остальные машины сети выполняют роль рабочих станций. Файлы БД по запросам пользователей передаются по сети на рабочие станции, где производится в основном обработка данных. Пользователи могут создавать на рабочих станциях локальные БД и пользоваться ими самостоятельно.

Архитектура Клиент-сервер предусматривает, что помимо хранения централизованной БД сервер базы данных должен обеспечивать выполнение основного объема обработки данных. По запросу клиента с рабочей станции система выполняет поиск и извлечение данных на сервере. Извлеченные данные (но не файлы) передаются по сети от сервера к клиенту.

При выборе СУБД для обработки БД к ним предъявляются следующие требования:

1. Непротиворечивость данных. Не должно возникнуть такой ситуации, когда заказывается отсутствующий на складе товар или в результате ошибки ввода информация о покупателе в заказе не соответствует данным картотеки покупателей. Такое требование называется требованием целостности. Целостность базы данных подразумевает систему правил, используемых в СУБД, для поддержания полной, непротиворечивой, и адекватно отражающей предметную область информации, обеспечивает защиту от случайного удаления или изменения данных в связанных таблицах. Целостность данных должна обеспечиваться независимо от того, каким образом данные заносятся в память (в интерактивном режиме, посредством импорта или с помощью специальной программы). С требованием целостности данных связано понятие транзакции. Транзакция – последовательность операций над БД, рассматриваемых как единое целое (то есть или все или ничего). Например, при оформлении заказа на определенный товар нужно выполнить такие

операции: регистрацию заказа на определенное количество товара и уменьшение количества данного товара на складе. Если на любом этапе изменения данных произойдет сбой, то целостность БД будет нарушена. Для исключения подобных нарушений вводится транзакция – операция «Оформление заказа», которая либо производит над БД все необходимые действия (товар продан, уменьшен его запас на складе), либо возвращает БД к исходному состоянию (товар не продан, его количество на складе не изменилось).

2. Актуальность данных. В любой момент времени информация БД должна быть современной и востребованной.

3. Многоаспектное использование данных. Возможность поступления информации в единую БД из различных источников и возможность ее использования любым отделом предприятия в соответствии с правами доступа и функциями.

4. Возможность модификации системы – возможность ее расширения и изменения данных, а также дополнение новыми функциями без ущерба для системы в целом.

5. Надежность и безопасность – целостность БД не должна нарушаться при технических сбоях. Средства обеспечения безопасности данных должны обеспечивать выполнение следующих операций: шифрование прикладных программ, шифрование данных, защиту паролем, ограничение уровня доступа.

6. Скорость доступа – обеспечение быстрого доступа к требуемой информации.

7. Импорт – экспорт данных – возможность обмена данными с другими программными средствами.

СУБД осуществляют взаимодействие между БД и пользователями системы, а также между БД и прикладными программами, выполняющими определенные функции обработки данных. К основным функциям СУБД относятся:

- Надежное хранение больших объемов данных сложной структуры во внешней памяти вычислительной системы.
- Непосредственное управление данными во внешней и оперативной памяти и обеспечение эффективного доступа к ним в процессе решения задач.
- Поддержание целостности данных и управление транзакциями.
- Обеспечение восстановления БД после технического и программного сбоя.
- Поддержка языка описания данных и языка запросов.
- Обеспечение безопасности данных.
- Обеспечение параллельного доступа к данным нескольких пользователей.

2. Основы проектирования баз данных.

В разработке ИС самым важным процессом является разработка проекта ее базы данных, так как допущенные на этом этапе ошибки в дальнейшем практически невозможно или очень сложно устранить. Проектирование базы данных заключается в ее многоступенчатом описании. Трем основным уровням моделирования системы – концептуальному, логическому и физическому соответствуют три последовательных этапа описания объектов БД и их взаимосвязей. В процессе проектирования БД производится описание информации предметной области, разработка, уточнение и оптимизация структуры БД с различной степенью детализации и формализации на разных этапах. Проектирование начинается с описания предметной области и задач ИС, идет к более абстрактному уровню логического описания данных и далее – к схеме физической (внутренней) модели БД.

На первом концептуальном этапе проектирования разрабатывается Концептуальная модель БД (от слова концепция – основное содержание), в которой на естественном языке с помощью диаграмм и других средств описываются моделируемые объекты предметной области, их свойства и взаимосвязи. Таким образом, на этом этапе выделяется и описывается информация, которая должна быть представлена в БД. Затем выбирается СУБД для реализации БД. Определяющими факторами при выборе СУБД являются вид программного продукта, категория пользователей (профессиональные программисты или конечные пользователи), простота использования, уровень коммуникационных средств (для

использования в сетях), стоимость. Концептуальная модель не зависит от конкретной выбранной СУБД и является основой для построения логической модели БД. Каждая СУБД работает с определенной моделью данных, под которой понимается способ взаимосвязи данных между собой: в виде иерархического дерева (иерархическая модель), сложной сетевой структуры (сетевая модель) или связанных таблиц. Большинство современных СУБД используют табличную модель данных, называемую реляционной.

На этапе логического проектирования выполняется преобразование данных концептуальной модели в логическую модель в рамках той модели данных, которую поддерживает выбранная СУБД (разрабатывается логическая структура БД). Логическая модель отражает информационное содержание и является основой для всех пользователей ИС (информационно-логическая модель БД). Она описывает всю БД как единое целое, не зависит от конкретной СУБД и может быть реализована на любой СУБД реляционного типа. Так как пользователи различаются по классу решаемых ими задач, то их делят на группы по правам доступа к определенным частям БД. Отдельное логическое представление данных для каждого пользователя называется внешней моделью данных или пользовательским представлением. На этапе физического проектирования разрабатывается физическая модель БД, производится выбор рациональной структуры хранения данных и методов доступа к ним, разработка программ и приложений, которые обеспечивает выбранная СУБД. На этом этапе решаются вопросы эффективного выполнения запросов к БД. Физическая модель зависит от конкретной выбранной СУБД и содержит информацию обо всех объектах БД (таблицах, процедурах, модулях, запросах и др.) и используемых типах данных. Одна и та же логическая модель может быть реализована в разных физических моделях. Физическое проектирование является начальным этапом реализации БД. Эксплуатация БД начинается с заполнения БД реальными данными. На этом этапе требуется сопровождение БД – проведение контроля целостности данных, непротиворечивости, резервное копирование, архивирование. В процессе использования БД выявляются недоработки, происходит уточнение, изменение требований к БД, возможно, принимается решение о ее модификации.

Одной из наиболее простых и популярных ИС является система управления кадрами (персоналом). Процесс проектирования БД рассмотрим на примере разработки фрагмента такой ИС.

3. Реляционный подход к организации данных в БД.

Обычно произвольная структура (модель) данных для удобства работы с ней преобразовывается в совокупность простых двумерных таблиц. Такое представление является наиболее удобным и для пользователя и для вычислительных систем – большинство современных ИС основаны на работе с такими таблицами. БД, которые состоят из двумерных таблиц называются **реляционными** (от англ. Relation – отношение). В теории проектирования БД таблица с данными называется **отношением**. Теория реляционных БД – это сложная математическая дисциплина, основы которой разработал в 70х годах XX века исследователь Э. Кодд (США). Основная терминология БД зависит от уровня описания (теория или практика), конкретного класса СУБД, от категории пользователя. Основную терминологию БД можно свести в следующую таблицу.

Таблица 1.

Теория БД	**Реляционные БД (практика)**
Отношение (ИО)	Таблица
Кортеж	Строка (Запись)
Домен (Атрибут)	Столбец (Поле)

Примечание. В теории БД совместно с термином Атрибут часто употребляют термин Домен атрибута. Домен – это множество значений данного атрибута (например, Дата рождения – это атрибут, а множество значений даты рождения для всех студентов группы – домен атрибута).

Основные свойства реляционных БД:

- Любые совокупности данных представляются в виде двумерных массивов – таблиц:

Таб_номер	Фамилия	Имя	Отчество	Дата_приема	Подразделение	Должность	Общий_стаж	Зарплата
101	Иванцев	Михаил	Петрович	12/30/99	отдел статистики	главный специалист	25	25000.00
102	Макаров	Сергей	Геннадьевич	5/23/99	отдел маркетинга	экономист	9	7500.00
103	Петрова	Анна	Владимировна	5/23/05	отдел статистики	специалист 1й категории	15	10000.00
104	Соколов	Андрей	Михайлович	1/16/05	отдел маркетинга	главный специалист	21	25000.00
105	Конаков	Олег	Викторович	7/15/04	отдел статистики	экономист	10	7500.00

- Каждая таблица состоит из фиксированного числа столбцов и некоторого (переменного) числа строк.
- Каждый столбец представляет конкретный атрибут (например, Табельный номер, Фамилия, Должность и др.). Столбцы таблицы называются полями. Для каждого поля определяются его характеристики:
  - уникальное имя поля;
  - тип поля (тип данных, хранящихся в этом поле);
  - размер, формат поля.
- Описание полей, составляемое разработчиком, называется структурой или макетом таблицы.
- Каждая строка таблицы называется записью. Система нумерует записи по порядку: 1,2,3,…n, где n – общее число записей в таблице на данный момент. Количество записей в процессе эксплуатации БД может меняться (от 0 до миллионов). Количество полей и их характеристики можно изменить с помощью специальной операции изменения макета (структуры) таблицы.
- Одно и то же поле может входить в состав нескольких таблиц одновременно.

В реляционных БД связи между таблицами осуществляются посредством ключевых полей (**первичных ключей**). **Первичный ключ** – это поле, значение которого однозначно определяет строку таблицы (запись). Первичные ключи используются для идентификации строк в таблице, ускорения поиска записей в таблице, связывания таблиц. Первичный ключ может быть **простым** – содержащим значения одного какого-либо поля таблицы (например, Табельный номер) или **составным** – содержащим значения нескольких полей (например, Фамилия и Дата рождения). Каждое значение ключевого поля в пределах таблицы должно быть **уникальным**, т.е. неповторяющимся. Указание первичного ключа – это единственный способ отличить один экземпляр объекта от другого (одну запись таблицы от другой). Обычно в качестве первичного ключа выбирается некоторый уникальный числовой идентификатор записи: *Код клиента, Код сотрудника, Код товара, Регистрационный номер, Табельный номер* и т.п. Для установления связи между двумя таблицами первичный ключ одной таблицы может добавляться в другую таблицу в качестве внешнего ключа.

Основной смысл реляционного подхода состоит в том, чтобы представить произвольную структуру данных в виде простой двумерной таблицы по правилам нормализации отношений. **Нормализация отношений** – это формальный аппарат правил и ограничений для формирования таблиц (отношений), который позволяет устранить дублирование данных, обеспечивает непротиворечивость хранимых в базе данных, уменьшает трудозатраты на ведение БД (ввод и корректировку информации). При формировании таблицы, используя правила нормализации, следует стремиться к исключению из нее полей, которые не связаны непосредственно с первичным ключом таблицы. Например, в таблицу ***Картотека*** можно включить *Код сотрудника* (или его табельный номер), *Фамилию*, *Код подразделения*, связанного с сотрудником и т.п. Но *Адрес подразделения* включать нельзя – это атрибут не сотрудника, а подразделения. *Адрес подразделения* должен находиться в той таблице, где первичным ключом является *Название подразделения* или его *Код*.

Для повышения производительности реляционные СУБД используют специальные объекты, называемые **индексами. Индекс** упорядочен по значению ключевого поля, что позволяет системе быстро находить нужные значения. Фактически индексная структура является «оглавлением». Но индексирование замедляет обновление записей. В реляционных СУБД таблицы всегда индексируются по полю (полям) первичного ключа. Можно создавать дополнительные индексы для ускорения поиска и выполнения основных запросов. Например, предполагается осуществлять поиск информации в БД по Фамилии сотрудника, то нужно предварительно создать индекс (провести индексирование таблицы) по полю Фамилия, который упорядочит записи в таблице по этому полю.

Одно из важнейших достоинств реляционных баз данных состоит в том, что можно хранить логически сгруппированные данные в разных таблицах и задавать **связи** между ними, объединяя их в единую базу. Для задания связи таблицы должны иметь поля с одинаковыми именами или хотя бы с одинаковыми форматами данных. Связь между таблицами устанавливает отношения между совпадающими значениями в этих полях. Такая организация данных позволяет уменьшать избыточность хранимых данных, упрощает их ввод и организацию запросов и отчетов.

В реляционных БД для указания взаимодействия между ИО (таблицами) используется три вида связей: **Один-ко-многим, Многие-ко-многим** и **Один-к-одному.**

Связь **Один-ко-многим** – наиболее часто используемый тип связи между таблицами. В такой связи каждой записи в таблице А может соответствовать несколько записей в таблице В, а запись в таблице В не может иметь более одной соответствующей ей записи в таблице А. Например, одному значению *должности* из таблицы ***Должности сотрудников*** соответствует несколько записей из таблицы ***Картотека***, содержащей анкетные данные работников предприятия (несколько сотрудников могут работать в одной и той же должности), и наоборот, каждому сотруднику соответствует только одно значение должности.

При связи **Один-к-одному** запись в таблице А может иметь не более одной связанной записи в таблице В и наоборот. Этот тип связи используют не очень часто, поскольку такие данные могут быть помещены в одну таблицу. Связь с отношением **Один-к-одному** применяют для разделения очень широких таблиц, для отделения части таблицы в целях ее защиты, а также для сохранения сведений, относящихся к подмножеству записей в главной таблице. Например, если анкетные данные на сотрудника хранятся в таблице ***Картотека***, а значение его выработки – в другой таблице ***Выработка***, то между этими таблицами устанавливается связь один-к-одному.

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

При проектировании реляционных таблиц рекомендуется придерживаться следующих правил:

- определить в таблице поле первичного ключа – убедиться, что записей с одинаковым значением ключа в таблице не будет;
- если первичный ключ не просматривается (его трудно определить), подумать, правильно ли подобран состав полей таблицы;
- если первичный ключ определен верно, к нему можно добавлять любые атрибуты, непосредственно зависящие от ключа;
- если при просмотре подготовленной БД обнаруживается в двух таблицах одноименное поле, которое не является первичным ключом ни в одной из этих таблиц, - это ошибка нормализации. Система не сможет контролировать согласованность таких полей.

Достоинства реляционной модели данных:

- Простота и доступность для понимания конечным пользователем.
- При проектировании реляционных БД применяются строгие правила, основанные на математическом аппарате.
- Независимость данных – при изменении структуры БД изменения, вносимые в прикладные программы, минимальны.
- Для построения запросов и написания прикладных программ нет необходимости знания конкретной организации БД во внешней памяти.

Недостатки реляционной модели.

- Реляционная модель БД имеет более низкую скорость доступа к данным по сравнению с другими моделями и требует большого объема внешней памяти вычислительной системы.
- Если в результате логического проектирования БД получается большое количество таблиц, то это затрудняет понимание структуры данных.
- Не всегда предметную область можно представить в виде набора таблиц.

1. 7 Лекция №7 (2 часа).

Тема: «Системы управления базами данных»

1.7.1 Вопросы лекции:

1. Общие свойства СУБД

2. Обобщенная схема обмена данных с использованием СУБД

3. Типовые информационные процедуры, реализуемые СУБД

4. Общие сведения о СУБД

1.7.2 Краткое содержание вопросов:

1. Общие свойства СУБД.

БД — это единое хранилище данных, которое однократно определяется, а после этого многократно используется разными пользователями для удовлетворения возникающих многообразных потребностей в информации.

ИС может быть

- одно- или
- многопользовательской.

Данные могут быть организованы в одну или несколько БД, данные в БД могут быть интегрированными (объединение нескольких отдельных файлов данных, полностью или частично не перекрывающихся) и общими (использование отдельных областей данных несколькими пользователями для разных целей).

Информация должна быть организована так, чтобы обеспечить минимальную избыточность. БД содержит информацию о данных, принятую называть «метаданными». В совокупности описания всех данных образуют словарь данных, обеспечивающий независимость данных от приложений. Данные должны быть логически связаны, для чего определяют объекты (то, о чем необходимо хранить информацию) и их свойства (простые и сложные), а затем выявляют связи между ними (то, что объединяет два или более объектов, они также являются частью данных и хранятся в БД).

Основные свойства БД:

- Целостность (В каждый момент времени существования БД сведения, содержащиеся в ней, должны быть непротиворечивы),
- Восстанавливаемость (Данное свойство предполагает возможность восстановления БД после сбоя системы или отдельных видов порчи системы),
- Безопасность (предполагает защиту данных от преднамеренного и непреднамеренного доступа, модификации или разрушения),
- Эффективность (сочетание минимального времени реакции на запрос пользователя и минимальной потребности в памяти),
- Предельные размеры и эксплуатационные ограничения.

К основным свойствам СУБД и базы данных можно отнести:

- отсутствие дублирования данных в различных объектах модели, обеспе̂чивающее однократный ввод данных и простоту их корректировки;
- непротиворечивость данных;
- целостность БД;
- возможность многоаспе̂ктного доступа;
- всевозможные выборки данных и их использование различными задачами и приложениями пользователя;
- защита и восстановление данных при аварийных ситуациях, аппаратных и программных сбоях, ошибках пользователя;

- защита данных от несанкционированного доступа средствами разграничения доступа для различных пользователей;

- возможность модификации структуры базы данных без повторной загрузки данных;

- обеспе̂чение независимости программ от данных, позволяющей сохранить программы при модификации структуры базы данных;

- реорганизация размещения данных базы на машинном носителе для улучшения объемно-временных характеристик БД;

- наличие языка запросов высокого уровня, ориентированного на конечного пользователя, который обеспе̂чивает вывод информации из базы данных по любому запросу и предоставление ее в виде соответствующих отчетных форм, удобных для пользователя.

СУБД является основой создания практических приложений пользователя для различных предметных областей.

Критерии выбора СУБД пользователем. Выбор СУБД для практических приложений пользователем определяется многими факторами, к которым относятся:

- имеющееся техническое и базовое программное обеспе̂чение, их конфигурация, опе̂ративная и дисковая память;

- потребности разрабатываемых приложений пользователя;

- тип поддерживаемой модели данных, спе̂цифика предметной области, топология информационно-логической модели;

- требования к производительности при обработке данных;

- наличие в СУБД необходимых функциональных средств;

- наличие русифицированной версии СУБД;

- уровень квалификации пользователей и наличие в СУБД диалоговых средств разработки и работ с БД.

СУБД — это программное обеспечение, с помощью которого пользователи могут определять, создавать и поддерживать базу данных, а также осуществлять к ней контролируемый доступ.

Различаются два класса СУБД — системы общего назначения (не ориентированы на какую-либо конкретную предметную область, обладают средствами настройки на работу с конкретной БД в условиях конкретного применения, реализуются как программный продукт, способный функционировать на некоторой модели ЭВМ в определенной операционной обстановке) и специализированные системы (весьма трудоемкий процесс, к которому прибегают, в исключительных ситуациях).

В процессе реализации своих функций СУБД постоянно взаимодействует с базой данных и с другими прикладными программными продуктами пользователя, предназначенными для работы с данной БД и называемыми приложениями.

СУБД включает язык определения данных, с помощью которого можно определить базу данных, ее структуру, типы данных, а также средства задания ограничений для хранимой информации.

СУБД позволяет вставлять, удалять, обновлять и извлекать информацию из базы данных посредством языка управления данными (язык запросов).

Большинство СУБД могут работать на компьютерах с разной архитектурой и под разными операционными системами.

Многопользовательские СУБД имеют достаточно развитые средства администрирования БД.

СУБД предоставляет контролируемый доступ к базе данных с помощью системы обеспечения безопасности, системы поддержки целостности базы данных, системы управления параллельной работой приложений, системы восстановления, доступного пользователям каталога, содержащего описание хранимой в базе данных информации.

Компоненты СУБД - это данные (наиболее важный компонентом среды СУБД, ради общения с ними на должном уровне требуется наличие остальных компонентов, они включают в себя имена, типы и размеры элементов данных, имена связей, ограничения

целостности данных, имена зарегистрированных пользователей и их права по доступу к данным, используемые индексы и структуры хранения), пользователи, аппаратное обеспечение (это набор физических устройств, на которых существуют база данных, СУБД и другие компоненты ИС, оно должно соответствовать требованиям использующей его организации, СУБД, БД. Это может быть один персональный компьютер или сеть), программное обеспечение (ОС, программное обеспечение самой СУБД, прикладные программы), процедуры.

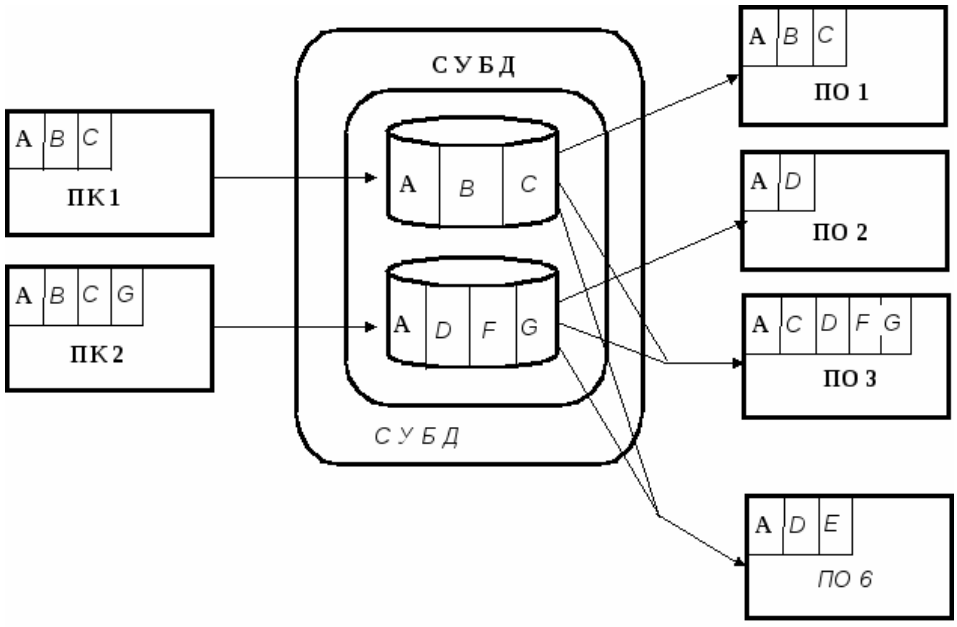
Среди пользователей БД можно выделить четыре категории лиц: администраторы данных (отвечают за концептуальное и логическое проектирование базы данных, управление данными, разработку и сопровождение стандартов, бизнес-правил и деловых процедур) и баз данных (отвечают за физическое проектирование и физическую реализацию базы данных, обеспечение безопасности и целостности данных, обеспечение максимальной производительности приложений), разработчики баз данных (группа пользователей, которая функционирует во время проектирования, создания и реорганизации базы данных и результатом деятельности которой является хорошо спроектированная БД, они делятся на разработчиков логической и физической БД, первые занимаются выявлением объектов, их свойств и связей между ними, вторые – разбираются в функциональных возможностях СУБД), прикладные программисты (разрабатывают приложения, предоставляющие пользователям необходимые им функциональные возможности), конечные пользователи (те, для кого проектируется, создается и поддерживается БД).

2. Обобщенная схема обмена данных с использованием СУБД.

СУБД — специализированные программные средства, предназначенные для организации и ведения баз данных.

СУБД используется для обеспечения эффективного доступа к базе данных со стороны программ (предоставление только необходимой информации, обеспечение независимости от возможных изменение в структуре той части базы данных, которую не обрабатывает программа), по существу она исполняет функции операционной системы по управлению данными. Схема иллюстрирует это положение.

В зависимости от способа взаимодействия СУБД и программы обработки либо в программу передается очередная запись требуемой структуры, не зависимо от того, где физически в БД расположены данные, составляющие требуемую структуру, либо для программы создается временный файл требуемой структуры.



1. СУБД включает язык определения данных, с помощью которого можно определить базу данных, ее структуру, типы данных, а также средства задания ограничений для хранимой информации.

2. СУБД позволяет вставлять, удалять, обновлять и извлекать информацию из базы данных посредством языка управления данными.

3. Большинство СУБД могут работать на компьютерах с разной архитектурой и под разными операционными системами, причем на работу пользователя тип платформы влияния не оказывает.

4. Многопользовательские СУБД имеют достаточно развитые средства администрирования БД.

5. СУБД предоставляет контролируемый доступ к базе данных с помощью:

системы обеспечения безопасности;

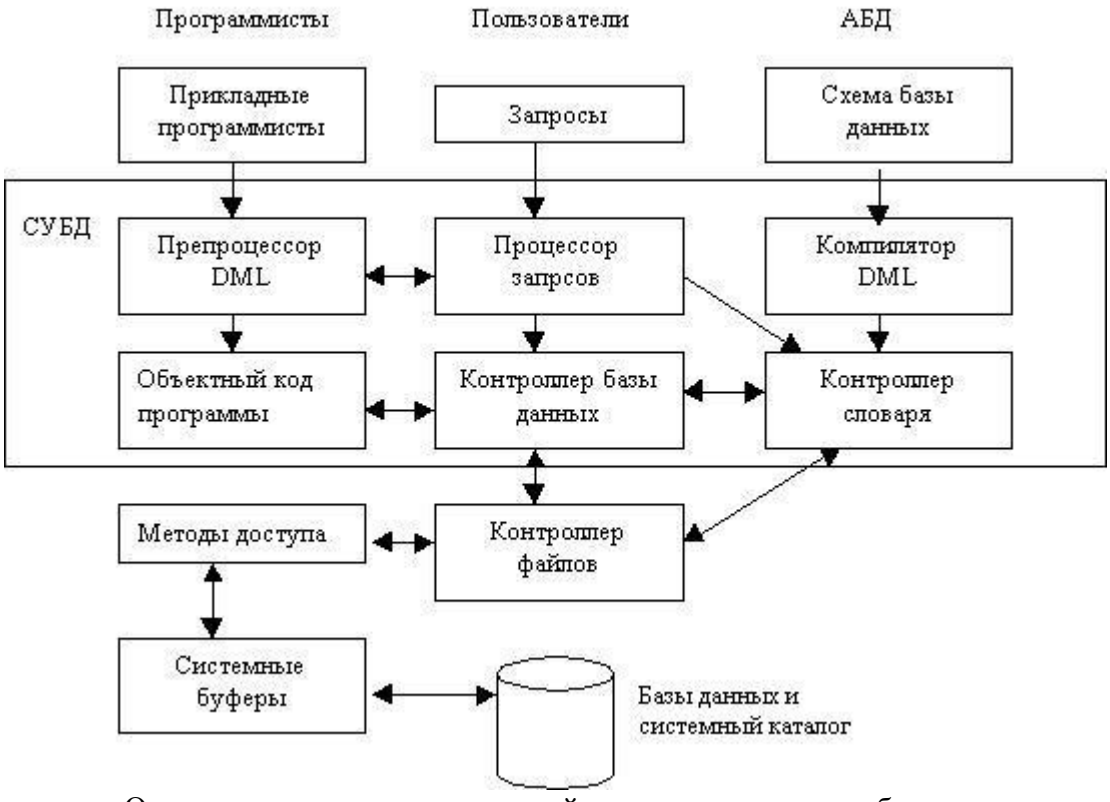
системы поддержки целостности базы данных;

системы управления параллельной работой приложений;

системы восстановления.

СУБД состоит из нескольких программных компонентов (модулей), каждый из которых предназначен для выполнения определенной операции. На схеме также показано, как СУБД взаимодействует с другими программными компонентами, например с такими, как пользовательские запросы и методы доступа (т.е. методы управления файлами, используемые при сохранении и извлечении записей с данными).

Процессор запросов. Это основной компонент СУБД, который преобразует запросы в последовательность низкоуровневых инструкций для контроллера базы данных.



Основные компоненты типичной системы управления базами данных

Контроллер базы данных. Этот компонент взаимодействует с запущенными пользователями прикладными программами и запросами. Контроллер базы данных принимает запросы и проверяет внешние и концептуальные схемы для определения тех концептуальных записей, которые необходимы для удовлетворения требований запроса.

Затем контроллер базы данных вызывает контроллер файлов для выполнения поступившего запроса.

Контроллер файлов манипулирует предназначенными для хранения данных файлами и отвечает за распределение доступного дискового пространства. Он создает и поддерживает список структур и индексов, определенных во внутренней схеме. Если используются хешированные файлы, то в его обязанности входит и вызов функций хеширования для генерации адресов записей. Однако контроллер файлов не управляет физическим вводом и выводом данных непосредственно, а лишь передает запросы соответствующим методам доступа, которые считывают данные в системные буферы или записывают их оттуда на диск.

Препроцессор языка DML. Этот модуль преобразует внедренные в прикладные программы DML-операторы в вызовы стандартных функций базового языка. Для генерации соответствующего кода препроцессор языка DML должен взаимодействовать с процессором запросов.

Компилятор языка DDL. Компилятор языка DDL преобразует DDL-команды в набор таблиц, содержащих метаданные. Затем эти таблицы сохраняются в системном каталоге, а управляющая информация - в заголовках файлов с данными.

Контроллер словаря. Контроллер словаря управляет доступом к системному каталогу и обеспечивает работу с ним. Системный каталог доступен большинству компонентов СУБД.

Ниже перечислены основные программные компоненты, входящие в состав контроллера базы данных.

Контроль прав доступа. Этот модуль проверяет наличие у данного пользователя полномочий для выполнения затребованной операции.

Процессор команд. После проверки полномочий пользователя для выполнения затребованной операции управление передается процессору команд.

Средства контроля целостности. В случае операций, которые изменяют содержимое базы данных, средства контроля целостности выполняют проверку того, удовлетворяет ли затребованная операция всем установленным ограничениям поддержки целостности данных (например, требованиям, установленным для ключей).

Оптимизатор запросов. Этот модуль определяет оптимальную стратегию выполнения запроса.

Контроллер транзакций. Этот модуль осуществляет требуемую обработку операций, поступающих в процессе выполнения транзакций.

Планировщик. Этот модуль отвечает за бесконфликтное выполнение параллельных операций с базой данных. Он управляет относительным порядком выполнения операций, затребованных в отдельных транзакциях.

Контроллер восстановления. Этот модуль гарантирует восстановление базы данных до непротиворечивого состояния при возникновении сбоев. В частности, он отвечает за фиксацию и отмену результатов выполнения транзакций.

Контроллер буферов. Этот модуль отвечает за перенос данных между оперативной памятью и вторичным запоминающим устройством - например, жестким диском или магнитной лентой. Контроллер восстановления и контроллер буферов иногда (в совокупности) называют контроллером данных.

Для воплощения базы данных на физическом уровне помимо перечисленных выше модулей нужны некоторые другие структуры данных. К ним относятся файлы данных и индексов, а также системный каталог. Группой DAFTG (Data-base Architecture Framework Task Group) была предпринята попытка стандартизации СУБД, и в 1986 году ею была предложена некоторая эталонная модель. Назначение эталонной модели заключается в определении концептуальных рамок для разделения предпринимаемых попыток стандартизации на более управляемые части и указания взаимосвязей между ними на очень широком уровне.

3. Типовые информационные процедуры, реализуемые СУБД.

Типовые информационные процедуры, реализуемые СУБД:

1. Определение структуры создаваемой базы данных, ее инициализация и проведение начальной загрузки. Данная процедура позволяет описать и создать в памяти структуру таблицы, провести начальную загрузку данных в таблицы.

2. Предоставление пользователям возможности манипулирования данными (выборка необходимых данных, выполнение вычислений, разработка интерфейса ввода/вывода, визуализация).Такие возможности в СУБД представляются либо на основе использования специального языка программирования, входящего в состав СУБД, либо с помощью графического интерфейса.

3. Обеспечение независимости прикладных программ и данных (логической и физической независимости).Обеспечение логической независимости данных предоставляет возможность изменения логического представления базы данных без необходимости изменения физических структур хранения данных. Обеспечение физической независимости данных предоставляет возможность изменять способы организации базы данных в памяти ЭВМ не вызывая необходимости изменения "логического" представления данных.

4. Защита логической целостности базы данных.

Основной целью реализации этой функции является повышение достоверности данных в базе данных. Достоверность данных может быть нарушена при их вводе в БД или при неправомерных действиях процедур обработки данных, получающих и заносящих в БД неправильные данные.

5. Защита физической целостности.

Развитые СУБД имеют средства восстановления базы данных, ведение журнала транзакций.

6. Управление полномочиями пользователей на доступ к базе данных.

7. Синхронизация работы нескольких пользователей.

Коллизии(одновременной обращение к одним данным) могут привести к нарушению логической целостности данных, поэтому система должна предусматривать меры, не допускающие обновление данных другим пользователям, пока работающий с этими данными пользователь полностью не закончит с ними работать.

8. Управление ресурсами среды хранения.

9. Поддержка деятельности системного персонала.

При эксплуатации базы данных может возникать необходимость изменения параметров СУБД, выбора новых методов доступа, изменения (в определенных пределах) структуры хранимых данных, а также выполнения ряда других общесистемных действий.

4. Общие сведения о СУБД.

Первые СУБД появились в конце 60-х годов, расцвет их применения приходится на 70-е начало 80-х годов. В то время на первый план выступала способность СУБД хранить данные сложной структуры и значительного объема и использовать установленные связи между информационными элементами при проектировании приложений. Другое важное достоинство - это обеспечение относительной независимости программ от структур хранения. Первые СУБД были ориентированы на программистов. Интерфейс с такими СУБД осуществлялся путем обращения к программе – СУБД из программы приложения, написанной на одном из базовых языков программирования и такие системы стали называть системами с базовым языком. При этом СУБД выполняла лишь простые операции выборки записей, удовлетворяющих определенным условиям и в определенной последовательности, а также включения, замены и удаления записей. Но все эти операции осуществлялись с учетом зафиксированной структуры БД, что существенно сокращало алгоритмическую часть программы, касающуюся согласованной выборки связанных записей, и снижало риск нарушения структурной целостности БД.

IMS (Information Management System) являлись весьма распространенными СУБД, обеспечивающими хранение и доступ к БД иерархической структуры.

Элементом структуры является сегмент, который может состоять из одного или нескольких полей. Поле может иметь тип: символьный или числовой. Сегменты одного типа имеют единую для этого типа внутреннюю структуру и размер, в том числе фиксированные размеры и типы одноименных полей в различных реализациях сегментов.

БД сложной логической структуры может быть разнесена в 10 физических наборов данных (файлов). IMS поддерживает 4 способа физической организации БД: иерархический последовательный метод доступа; иерархический индексно-последовательный метод доступа; иерархический индексно-прямой метод доступа и иерархический прямой.

Система IMS(ОКА) является системой с базовым языком.

Единицей обмена между прикладной программой и БД является реализация сегмента определенного типа (с подсоединенными ключами старших сегментов) или совокупности реализаций сегментов, расположенных в одной иерархической ветви.

Система обеспечивает выполнение всех типичных для СУБД функций: вставку, замену, удаление и чтение реализаций сегментов.

Система ID

СУБД IDS обеспечивают хранение и доступ к БД с сетевой структурой.

Основной структурной единицей, обеспечивающей в конечном счете сетевую структуру любой сложности, является цепь. В цепь объединяются информационно зависимые логические записи.

Запись представляет собой основную единицу информации и состоит из полей данных и служебных полей.

Служебные поля предназначены для идентификации записи и организации цепей с помощью адресных ссылок.

Информационные поля могут содержать числовые и символьные данные и имеют имена.

Записи одного типа содержат одноименные данные и имеют фиксированную длину, равно как и одноименные поля. Записям одного типа присваиваются уникальное имя и тип. Записи адресуются в БД с помощью адресных ссылок, состоящих из номера «страницы» БД и номера байта начала записи на «странице». Цепи образуются с помощью адресных ссылок, хранящихся в записи и указывающих на следующую запись в этой цепи. Каждый цепи присваивается уникальное имя. Каждая запись может входить в любое количество цепей.

Система также обеспечивает типовые операции доступа к данным – запоминание, выборку, модификацию и удаление записей из программ, написанных на классических языках программирования.

Система ADABAS

ADABAS сочетает в себе возможности СУБД с базовым языком и системы замкнутого типа и обеспечивает поддержание ограниченных сетевой и иерархической структур. БД в ADABAS является совокупностью связанных данных, организованных в виде файлов. Реализованное в системе динамическое установление связей между записями различных файлов позволяет создавать межзаписные иерархические и сетевые структуры. ADABAS осуществляет поиск по запросам пользователей записей в БД. Доступ к хранимым данным, обработку данных и выдачу результатов в виде справок и сводок заданной формы. Запросы вводятся в диалоговом режиме с видеотерминалов и в пакетном режиме в образе перфокарт. Языки запросов высокого уровня ориентированы на пользователей-непрограммистов. Обеспечивается и мультидоступ к БД.

ADABAS дает возможность обращаться к БД и из прикладных программ, написанных на АССЕМБЛЕРЕ, КОБОЛе, ФОРТРАНе и PL/1, в том числе под управлением телемонитора.

Система FoxPro

СУБД FoxPro обеспечивает возможность работы в трех режимах:

• выполнение откомпилированных программ, написанных на языке СУБД (xBASE-язык);

• выполнение команд языка xBASE или команд SQL в режиме интер-претации;

• обработка баз данных в экранном интерфейсе (без предварительного программирования и без знания команд языка xBASE).

База данных FoxPro состоит из совокупности взаимосвязанных файлов, каждый из которых может быть представлен в виде таблицы. Запись файла соответствует строке таблицы, поле – столбцу таблицы. Файл и поля имеют имена – идентификаторы назначаемые при первоначальном создании файла.

Все записи одного файла однотипны, имеют фиксированную длину, потому как одноименные поля во всех записях имеют один и тот же размер и тип значения. В одной записи (строке таблицы) у одного поля может быть только единственное значение. В файле (таблице) не может быть двух одинаковых записей. Одно либо несколько полей являются ключом записи. Значение ключа уникально.

Файлы могут быть связаны друг с другом:

• Возможные виды связей 1 : 1, 1 : M, M : 1, M : N:

Система Access

СУБД Access является СУБД реляционного типа, в которой разумно сбалансированы все средства и возможности, типичные для современных СУБД. В отличии от FoxPro, в Access все данные вместе с запросами и формами физически хранятся в одном файле.

База данных Microsoft Access состоит из следующих объектов:

- таблицы,
- запросы,
- формы,
- отчеты,
- макросы,
- модули.

Поскольку, как правило, система управления базами данных обрабатывает одновременно несколько таблиц, то существует возможность установления реляционных связей между таблицами.

Клиент-сервер

При использовании клиент-серверной технологии, на самом сервере, содержащим базу данных, функционирует некоторое программное обеспечение, которое называется "Сервером баз данных" или "Сервером БД". Клиент-серверная СУБД позволяет обмениваться клиенту и серверу минимально необходимыми объёмами информации. При этом основная вычислительная нагрузка ложится на сервер. Клиент может выполнять функции предварительной обработки перед передачей информации серверу, но в основном его функции за-ключаются в организации доступа пользователя к серверу. Таким образом, архитектура клиент-сервер адаптирована для работы с большими объемами данных - сеть нагружается меньше, требования к пользовательским компьютерам, с точки зрения производительности, минимизируется. Однако возрастают требования к серверу, содержащему базу данных, поскольку теперь он один тянет нагрузку всех пользователей. Клиент-серверная СУБД располагается на сервере вместе с БД и осуществляет доступ к БД непосредственно, в монопольном режиме. Все клиентские запросы на обработку данных обрабатываются клиент-серверной СУБД централизованно. Недостаток клиент-серверных СУБД состоит в повышенных требованиях к серверу. Достоинства: потенциально более низкая загрузка локальной сети; удобство централизованного управления; удобство обеспечения таких важных характеристик как высо-кая надёжность, высокая доступность и высокая безопасность.

Примеры: Oracle, Firebird, Interbase, IBM DB2, Informix, MS SQL Server, Sybase Adaptive Server Enterprise, PostgreSQL, MySQL, Cache.

1. 8 Лекция №8 (2 часа).

Тема: «Создание и модификация базы данных»

1.8.1 Вопросы лекции:

1. Проектирование с использованием метода сущность – связь
2. Традиционные методики проектирования БД
3. Современная интеграционная методика проектирования
4. Проектирование системы баз данных на принципах единой информационной среды

1.8.2 Краткое содержание вопросов:

1. Проектирование с использованием метода сущность – связь.

Метод "сущность–связь" (entity–relation, ER–method) является комбинацией двух предыдущих и обладает достоинствами обоих. Этап инфологического проектирования начинается с моделирования ПО. Проектировщик разбивает её на ряд локальных областей, каждая из которых (в идеале) включает в себя информацию, достаточную для обеспечения запросов отдельной группы будущих пользователей или решения отдельной задачи (подзадачи). Каждое локальное представление моделируется отдельно, затем они объединяются.

Выбор локального представления зависит от масштабов ПО. Обычно она разбивается на локальные области таким образом, чтобы каждая из них соответствовала отдельному внешнему приложению и содержала 6-7 сущностей.

Сущность – это объект, о котором в системе будет накапливаться информация. Сущности бывают как физически существующие (например, СОТРУДНИК или АВТОМОБИЛЬ), так и абстрактные (например, ЭКЗАМЕН или ДИАГНОЗ).

Для сущностей различают тип сущности и экземпляр. Тип характеризуется именем и списком свойств, а экземпляр – конкретными значениями свойств.

Типы сущностей можно классифицировать как сильные и слабые. Сильные сущности существуют сами по себе, а существование слабых сущностей зависит от существования сильных. Например, читатель библиотеки – сильная сущность, а абонемент этого читателя – слабая, которая зависит от наличия соответствующего читателя. Слабые сущности называют подчинёнными (дочерними), а сильные – базовыми (основными, родительскими).

Для каждой сущности выбираются свойства (атрибуты). Различают:

- Идентифицирующие и описательные атрибуты. Идентифицирующие атрибуты имеют уникальное значение для сущностей данного типа и являются потенциальными ключами. Они позволяют однозначно распознавать экземпляры сущности. Из потенциальных ключей выбирается один первичный ключ (ПК). В качестве ПК обычно выбирается потенциальный ключ, по которому чаще происходит обращение к экземплярам записи. Кроме того, ПК должен включать в свой состав минимально необходимое для идентификации количество атрибутов. Остальные атрибуты называются описательными и заключают в себе интересующие свойства сущности.

- Составные и простые атрибуты. Простой атрибут состоит из одного компонента, его значение неделимо. Составной атрибут является комбинацией нескольких компонентов, возможно, принадлежащих разным типам данных (например, ФИО или адрес). Решение о том, использовать составной атрибут или разбивать его на компоненты, зависит от характера его обработки и формата пользовательского представления этого атрибута.

- Однозначные и многозначные атрибуты (могут иметь соответственно одно или много значений для каждого экземпляра сущности).

- Основные и производные атрибуты. Значение основного атрибута не зависит от других атрибутов. Значение производного атрибута вычисляется на основе значений других атрибутов (например, возраст студента вычисляется на основе даты его рождения и текущей даты).

- Спецификация атрибута состоит из его названия, указания типа данных и описания ограничений целостности – множества значений (или домена), которые может принимать данный атрибут.

Далее осуществляется спецификация связей внутри локального представления. Связи могут иметь различный содержательный смысл (семантику). Различают связи типа "сущность-сущность", "сущность-атрибут" и "атрибут-атрибут" для отношений между атрибутами, которые характеризуют одну и ту же сущность или одну и ту же связь типа "сущность-сущность".

Каждая связь характеризуется именем, обязательностью, типом и степенью. Различают факультативные и обязательные связи. Если вновь порождённый объект одного типа оказывается по необходимости связанным с объектом другого типа, то между этими типами объектов существует обязательная связь (обозначается двойной линией). Иначе связь является факультативной.

По типу различают множественные связи "один к одному" (1:1), "один ко многим" (1:n) и "многие ко многим" (m:n). ER–диаграмма, содержащая различные типы связей, приведена на рис. 1. Обратите внимание, что обязательные связи на рис. 1 выделены двойной линией.

Рис.1. ER–диаграмма с примерами типов множественных связей

Степень связи определяется количеством сущностей, которые охвачены данной связью. Пример бинарной связи – связь между отделом и сотрудниками, которые в нём работают. Примером тернарной связи является связь типа экзамен между сущностями ДИСЦИПЛИНА, СТУДЕНТ, ПРЕПОДАВАТЕЛЬ. Из последнего примера видно, что связь также может иметь атрибуты (в данном случае это Дата проведения и Оценка). Пример ER–диаграммы с указанием сущностей, их атрибутов и связей приведен на рис. 2.

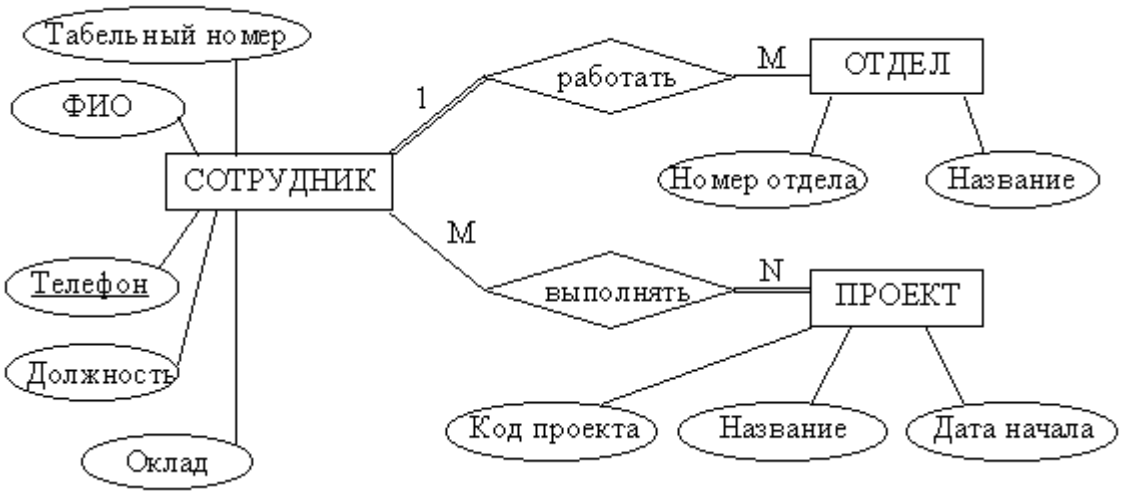


Рис.2. Пример ER–диаграммы с однозначными и многозначными атрибутами

После того, как созданы локальные представления, выполняется их объединение. При небольшом количестве локальных областей (не более пяти) они объединяются за один шаг. В противном случае обычно выполняют бинарное объединение в несколько этапов.

При объединении проектировщик может формировать конструкции, производные по отношению к тем, которые были использованы в локальных представлениях. Такой подход может преследовать следующие цели:

- объединение в единое целое фрагментарных представлений о различных свойствах одного и того же объекта;
- введение абстрактных понятий, удобных для решения задач системы, установление их связи с конкретными понятиями, использованными в модели;
- образование классов и подклассов подобных объектов (например, класс "изделие" и подклассы типов изделий, производимых на предприятии).

На этапе объединения необходимо выявить и устранить все противоречия. Например, одинаковые названия семантически различных объектов или связей или несогласованные ограничения целостности на одни и те же атрибуты в разных приложениях. Устранение противоречий вызывает необходимость возврата к этапу моделирования локальных представлений с целью внесения в них соответствующих изменений.

По завершении объединения результаты проектирования являют собой концептуальную инфологическую модель предметной области. Модели локальных представлений – это внешние инфологические модели.

2. Традиционные методики проектирования БД.

Методология проектирования, включающая в себя совокупность теоретических и инженерных знаний, обеспечивает упорядоченное создание больших информационных систем. Благодаря эффективной методологии проектирование информационных систем становится процессом создания промышленных, строго регламентированных изделий.

Методология проектирования включает три части:

- основные концепции и понятия, используемые при проектировании и реализации систем;
- технологию, организацию и управление процессом проектирования;
- инструментальные средства.

Попытка сформулировать общие требования к методологиям проектирования была предпринята в работе.

Главным требованием является охват методологией как можно, большего числа этапов жизненного цикла системы, которые предусматривают: оценку целей и возможностей создания системы, анализ требований; детальное проектирование; программирование и тестирование; интеграцию с существующей системой; внедрение и поддержку.

Важным требованием любой методологии является взаимосвязь этапов. Должна обеспечиваться связь с другими проектами, например преемственность стандартов.

Методология проектирования должна обеспечивать представление семантических требований к создаваемой системе и минимизировать потери информации при переходе от одного семантического уровня представления к другому. В этой связи большое значение имеют средства спецификации, используемые на различных уровнях представления данных.

Под спецификацией понимается точное, полное описание требований, ясно сформулированное в терминах, характерных для целей данной задачи, а не для ее реализации. Спецификации должны использоваться не только при проектировании программ и данных, но и при контроле их противоречивости.

Главное различие между методологиями, как правило, заключается в том, каким способом и в результате каких технологических операций может быть специфицирован проект создаваемой информационной системы.

Современные методологии проектирования ИС должны обеспечивать представление следующей информации:

- описание объекта автоматизации, а также места разрабатываемой информационной системы и целей, которые должны быть достигнуты в процессе разработки системы;

- описание функциональных возможностей ИС, достаточное для решения вопроса о том, что поставленные цели автоматизации достижимы;

- спецификации проекта, гарантирующие достижение заданных технических характеристик системы;

- описание реализации предлагаемой системы, достаточное для оценки времени разработки системы и необходимых для этой цели трудозатрат;

- детальный план создания системы с оценкой сроков разработки.

Таким образом, современная методология проектирования должна поддерживать сбор данных, их анализ, проектирование, оценку проекта и оценку возможности удовлетворения техническим характеристикам разрабатываемой системы.

Повышение производительности разработки программного обеспечения информационных систем достигается в основном за счет применения более совершенных системных программных средств и методологий проектирования.

Эволюция информационных систем от пакетной, массовой технологии обработки данных к системам, использующим базы данных, выявила три класса наиболее перспективных методологий проектирования. Первый из них ориентирован на концептуальное моделирование предметной области и технологию баз данных, второй — на выявление требований и спецификацию информационной системы через ее макетирование, третий — на системную архитектуру программных средств, поддерживаемую инструментальными средствами CASE (Computer Aided System Engineering) — технологии.

Методология проектирования информационных систем на основе концептуального (понятийного) моделирования предметной области (ПО) — одна из наиболее часто используемых. Она представляет собой структурированный процесс создания систем, который обычно разбивается на следующие шаги: анализ, проектирование, программирование, тестирование и внедрение.

При концептуальном моделировании ПО и применении технологии БД наиболее сложной задачей является выявление информационных и функциональных (динамических) связей между объектами реального мира.

Информационная структура ПО содержит все объекты и их связи, которые необходимы для построения ИС, а функциональная структура определяет, каким образом используются и обрабатываются эти объекты. Информационная и функциональная структуры совместно обеспечивают полную спецификацию информационной системы.

Создание ДИС на основе методологии концептуального проектирования предполагает четыре этапа проектирования:

- сбор и анализ информационных потребностей пользователей и системный анализ предметной области;

- построение концептуальной (понятийной) модели предметной области;

- создание концептуальной модели базы данных;

- разработку системы с помощью инструментальных средств выбранной СУБД.

Первый очень важный этап разработки системы — анализ требований может быть определен как этап понимания задач приложений. Именно здесь пользователи будущей системы должны сформулировать, а разработчики понять, что должна делать система, какие у нее особенности, какие понятия используются в задачах, какие ситуации предметной области должны моделироваться в базе данных.

На втором этапе разработчики системы должны определить устойчивые свойства данных и описать информационные и технологические процессы, использующие данные, их взаимосвязь и характеристики. Иногда эту работу определяют как построение концептуальной (инфологической) модели предметной области, содержащей описание понятий, не ориентированное ни на какую конкретную даталогическую модель.

Концептуальная модель предметной области ориентирована на восприятие человека (пользователя и разработчика), а не на обработку данных в ЭВМ. Именно с помощью этой модели разработчики ИС (да и сами пользователи) достигают высокого уровня понимания существа информационных потребностей пользователей. В настоящее время для построения концептуальной модели предметной области обычно используют два подхода. При первом подходе модель ПО строится на основе интеграции спецификаций информационных потребностей, а при втором — на основе непосредственного анализа самой ПО.

В первом случае более широко применяются инструментальные средства системных программных продуктов, которые интегрируются в единую программную систему, обеспечивающую обработку доступной информации как на этапе анализа, так и на этапе проектирования прикладной системы.

В методологиях проектирования, основанных на непосредственном создании концептуальной модели предметной области, основной задачей является получение формального (независимого от СУБД) описания предметной области, которая должна моделироваться в БД. При этом система проектирования и методология проектирования должны поддерживать как получение от пользователей знаний о свойствах предметной области, так и отображение этих упорядоченных и организованных знаний в набор предварительных описаний, составляющих собственно концептуальную модель предметной области.

Концептуальная модель включает описание понятий предметной области и информационных процессов, протекающих в ней, т. е. содержит всю необходимую для проектирования системы информацию. Информации должно быть достаточно, чтобы принимать правильные решения при проектировании не только БД, но и программной реализации задач.

Однако не все понятия и операции, используемые при описании в рамках информационных процессов ПО, должны моделироваться в БД системы. Ряд семантических преобразований данных осуществляется при реализации приложений. Например, такие преобразования могут выполняться задачами, обеспечивающими специфическую обработку информации для формирования выходных отчетов. Поэтому основной проблемой третьего этапа является принятие решения о выделении из множества понятий концептуальной модели предметной области таких объектов, которые должны моделироваться в БД.

На первых трех этапах проводится не зависящий от технических и системных программных средств анализ целей и назначения проектируемой ДИС и моделируются основные информационные и технологические процессы ее функционирования. Результаты, полученные на этих этапах, имеют фундаментальный характер и не изменяются при развитии технической и программной базы ИС. Напротив, заключительный этап проектирования тесно связан с возможностями инструментальных средств конкретных СУБД.

Данный этап в свою очередь разбивают на следующие шаги:

логическое проектирование БД;

физическое проектирование БД;

реализация приложений.

В соответствии с вышеизложенным методология проектирования ИС на основе концептуального моделирования ПО может быть представлена в виде последовательности этапов, изображенных на рис. 1.4.

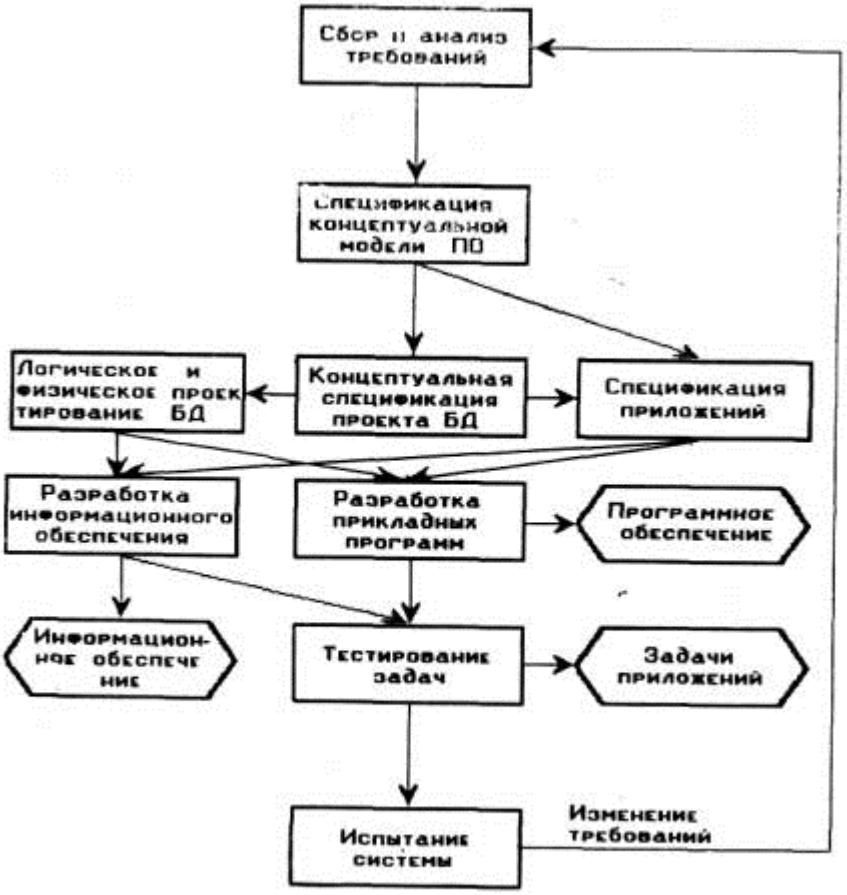


Рис. 1.4. Этапы проектирования ИС на основе концептуального моделирования ПО

Основу данной методологии составляет жестко регламентированная последовательность шагов, каждый из которых должен завершаться формальной спецификацией вне зависимости от того, насколько точно сформулированы спецификации предыдущих этапов проектирования. Такая стратегия проектирования обычно гарантирует разработку удовлетворительной системы при условии, что требования к системе фиксируются и их изменение не допускается до тех пор, пока система не предъявляется пользователям. Однако в этом случае ошибки проектирования могут быть обнаружены только на последних стадиях проектирования (либо при тестировании системы, либо во время проведения ее испытаний), что требует пересмотра всех спецификаций проекта и ведет к значительным трудозатратам.

Для поддержания методологии проектирования на основе концептуального моделирования ПО широко применяются языки спецификаций, непроцедурные языки программирования, инструментальные средства отладки программ и т. д. Однако возможность повышения производительности проектирования ИС при данном подходе остается ограниченной, так как на этапе внедрения часто выясняется, что важные требования к ИС остались неучтенными в созданной системе, а исправление допущенных ошибок требует больших трудозатрат.

Среди наиболее крупных работ в области концептуального моделирования, появившихся в последнее время, можно отметить модели Броуди, Кинга и Милополуоса, проект DATAID, где рассмотрены три модели концептуального моделирования, а также модель, используемую в методике REMORA. Значительным теоретическим достижением является также проект DIADA, в котором предложена трехуровневая структура инструментальных средств проектирования интерактивных информационных систем с интенсивным использованием данных.

Особенность указанных методологий заключается в интегрированном рассмотрении понятий, используемых для моделирования структурных и поведенческих свойств объектов реального мира.

Структурные свойства описываются с помощью абстракций типизации, агрегации, обобщения и ассоциации, а для описания поведенческих аспектов чаще всего употребляется концепция события.

Абстракции обеспечивают четкое, структурированное представление об объектах предметной области, их классах и взаимосвязях между объектами и классами объектов.

События, отражая факт изменения происшедшего в одном или нескольких объектах, позволяют адекватно учесть динамические процессы, происходящие в ПО.

3. Современная интеграционная методика проектирования.

Направление интегрированных или федеративных систем неоднородных БД и мульти-БД появилось в связи с необходимостью комплексирования систем БД, основанных на разных моделях данных и управляемых разными СУБД.

Основной задачей интеграции неоднородных БД является предоставление пользователям интегрированной системы глобальной схемы БД, представленной в некоторой модели данных, и автоматическое преобразование операторов манипулирования БД глобального уровня в операторы, понятные соответствующим локальным СУБД. В теоретическом плане проблемы преобразования решены, имеются реализации.

При строгой интеграции неоднородных БД локальные системы БД утрачивают свою автономность. После включения локальной БД в федеративную систему все дальнейшие действия с ней, включая администрирование, должны вестись на глобальном уровне. Поскольку пользователи часто не соглашаются утрачивать локальную автономность, желая тем не менее иметь возможность работать со всеми локальными СУБД на одном языке и формулировать запросы с одновременным указанием разных локальных БД, развивается направление мульти-БД. В системах мульти-БД не поддерживается глобальная схема интегрированной БД и применяются специальные способы именования для доступа к объектам локальных БД. Как правило, в таких системах на глобальном уровне допускается только выборка данных. Это позволяет сохранить автономность локальных БД.

Как правило, интегрировать приходится неоднородные БД, распределенные в вычислительной сети. Это в значительной степени усложняет реализацию. Дополнительно к собственным проблемам интеграции приходится решать все проблемы, присущие распределенным СУБД: управление глобальными транзакциями, сетевую оптимизацию запросов и т.д. Очень трудно добиться эффективности.

Как правило, для внешнего представления интегрированных и мульти-БД используется (иногда расширенная) реляционная модель данных. В последнее время все чаще предлагается использовать объектно-ориентированные модели, но на практике пока основой является реляционная модель. Поэтому, в частности, включение в интегрированную систему локальной реляционной СУБД существенно проще и эффективнее, чем включение СУБД, основанной на другой модели данных.

4. Проектирование системы баз данных на принципах единой информационной среды.

Говоря о единстве информационных процессов мы должны понимать, что информационные системы как таковые окружают нас повсюду. Более того, мы практически всегда имеем возможность найти основания для их концептуального объединения. Например, сетевая операционная система локальной вычислительной сети, функционирующая в рамках одного факультета университета, является объединяющей информационной системой для каждой отдельной инфосистемы – операционной системы персонального компьютера. Факультетская же сеть, в свою очередь, является элементом общей университетской локальной сети. Другой подход, акцентирующий программную

составляющую системной логики, позволяет нам рассматривать в качестве инфосистемы контроллер домена, с установленной на нем операционной системой Windows Server, и, соответственно, единой инфосистемой будет вся структура домена – комплекс входящих в него компьютеров под управлением главного сервера. Таким образом, мы видим, что различия в подходах к определению единой информационной системы связаны с различными основаниями для объединения отдельных инфосистем в единую сеть.

Один из авторов предлагает под единой средой взаимодействия (ЕСВ) понимать «совокупность технических и программных средств учреждения, реализующих идеи и методы автоматизации в сфере делового общения. Комплексная автоматизация подразумевает перевод в плоскость компьютерных технологий все основные деловые процессы организации (из которых нас интересует как раз сфера делового общения). Современные системы управления деловыми процессами позволяют интегрировать вокруг себя различное программное обеспечение, формируя единую информационную систему".

Из приведенного определения уже можно определить основную целеустановку ЕСВ в организации, либо преобразования существующих разрозненных систем в единую. Это автоматизация учета и обработки информационных потоков в рамках объединенной инфосистемы.

В крупных организациях и на предприятиях для создания единой информационной системы используют системы ERP (Enterprise Resource Planning). ERP-системы представляют собой набор интегрированных приложений, которые позволяют создать единую среду для автоматизации планирования, учета, контроля и анализа всех основных бизнес-операций в масштабе предприятия.

Единая среда взаимодействия – это информационная система, проектируемая согласно логической модели, отражающей цели ее создания. Технологически логическая модель системы создается на основе CMS (системы управления веб-контентом) и размещается в интернете.

Использование CMS Joomla (на основе лицензии GNU) практически не имеет аналогов по соотношению «цена-функциональность-удобство».

CMS Joomla представляет собой объектно-ориентированную модульную систему, написанную на скриптовом языке PHP. Удобный, модщный и, одновременно, несложный для освоения центр управления позволяет создавать веб-порталы практически любого уровня сложности.

Создание новых разделов осуществляется из центра управления. В дальнейшем, создавать и редактировать содержание разделов и модулей (форумов, блогов и т. д.) могут пользователи системы согласно присвоенным им правам доступа.

Содержание портала можно представить в виде иерархически распределенных документов с возможностью коллективного их редактирования (модерирования), реализовывая, таким образом, механизм wiki.

Единая регистрация пользователя в системе позволит ему получить доступ (согласно его уровню полномочий) сразу ко всем ресурсам системы – различным площадкам для общения, файловым хранилищам и т. д. Вся информация (и весь контент, и все способы представления данных – шаблоны) сохраняется в единой базе данных, что позволяет легко архивировать данные и восстанавливать всю систему в случае аварийного сбоя.

1. 9 Лекция №9 (2 часа).

Тема: «Информационные системы, основанные на БД и СУБД»

1.9.1 Вопросы лекции:

1. Обобщенная схема информационной системы, основанной на БД и СУБД
2. Состав и функции средств актуализации БД, средств обработки БД в интересах пользователей, средств администрирования БД
3. Технологии файл-сервер и клиент-сервер

1.9.2 Краткое содержание вопросов:

1. Обобщенная схема информационной системы, основанной на БД и СУБД.

Информационная система – это любая система, реализующая или поддерживающая информационный процесс.

К информационным можно относить любые системы, включающие в себя работу с информацией. В настоящее время основным помощником человека при работе с информацией является компьютер, поэтому именно его мы и будем рассматривать в качестве источника, способа изменения и хранения информационных систем. А в качестве информационных систем будем рассматривать программное обеспечение компьютера.

В зависимости от предметной области информационные системы могут весьма значительно различаться по своим функциям, архитектуре, реализации. Однако можно выделить ряд свойств, которые являются общими.

Информационные системы предназначены организации и поддержке информационного процесса, поэтому в основе любой из них лежит среда хранения и доступа к информации.

Информационные системы ориентированы на конечного пользователя, не обладающего высокой квалификацией в области вычислительной техники. Поэтому клиентские приложения информационной системы должны обладать простым, удобным, легко осваиваемым интерфейсом.

Таким образом, при разработке информационной системы приходится решать две основные задачи:

- разработка базы данных, предназначенной для хранения информации;
- разработка графического интерфейса пользователя клиентских приложений.

Подавляющее большинство информационных систем работает в режиме диалога с пользователем.

В наиболее общем случае типовые программные компоненты, входящие в состав информационной системы, реализуют:

- диалоговый ввод-вывод;
- логику диалога;
- прикладную логику обработки данных;
- логику управления данными;
- операции манипулирования файлами и (или) базами данных.

Классификация информационных систем

Информационные системы классифицируются по разным признакам:

1. Классификация по масштабу (рис1)

По масштабу информационные системы подразделяются на следующие группы:

- одиночные;
- групповые;
- корпоративные.

Рис. 1. Деление информационных систем по масштабу.

Одиночные информационные системы реализуются, как правило, на автономном персональном компьютере (сеть не используется). Такая система может содержать несколько простых приложений, связанных общим информационным фондом, и рассчитана на работу одного пользователя или группы пользователей, разделяющих по времени одно рабочее место. Подобные приложения создаются с помощью так называемых настольных, или локальных, систем управления базами данных (СУБД). Среди локальных СУБД наиболее известными являются Clarion, Clipper, FoxPro, Paradox, dBase и Microsoft Access.

Групповые информационные системы ориентированы на коллективное использование информации членами рабочей группы и чаще всего строятся на базе локальной вычислительной сети. При разработке таких приложений используют-ся серверы баз данных (называемые также SQL (Structured Query Language – структурированный язык запросов)-серверами) для рабочих групп. Существует довольно большое количество различных SQL-серверов как коммерческих, так и свободно распространяемых. Среди них наиболее известны такие серверы баз данных, как Oracle, DB2, Microsoft SQL Server, InterBase, Sybase, Informix.

Корпоративные информационные системы являются развитием систем для рабочих групп, они ориентированы на крупные компании и могут поддерживать тер-риториально разнесенные узлы или сети. В основном они имеют иерархическую структуру из нескольких уровней. Для таких систем характерна архитектура клиент-сервер со специализацией серверов или же многоуровневая архитектура. При разработке таких систем могут использоваться те же серверы баз данных, что и при разработке групповых информационных систем. Однако в крупных информационных системах наибольшее распространение получили серверы Oracle, DB2 и Microsoft SQL Server.

2. Классификация по сфере применения

По сфере применения информационные системы обычно подразделяются на четыре группы (рис. 2):

- системы обработки транзакций (протоколов);
- системы поддержки принятия решений;
- информационно-справочные системы;
- офисные информационные системы.

Рис. 2. Деление информационных систем по сфере применения.

Системы обработки транзакций, в свою очередь, по оперативности обработки данных разделяются на пакетные информационные системы и оперативные информационные системы. В информационных системах организационного управления преобладает режим оперативной обработки транзакций (OnLine Transaction Pro-cessing, OLTP) для отражения актуального состояния предметной области в любой момент времени, а пакетная обработка занимает весьма ограниченную часть. Для систем OLTP характерен регулярный (возможно, интенсивный) поток довольно простых транзакций, играющих роль заказов, платежей, запросов и т.п. Важными требованиями для них являются:

- высокая производительность обработки транзакций;
- гарантированная доставка информации при удаленном доступе к БД по телекоммуникациям.

Системы поддержки принятия решений (Decision Support System, DSS) представляют собой другой тип информационных систем, в которых с помощью довольно сложных запросов производится отбор и анализ данных в различных разрезах: временных, географических, по другим показателям.

Обширный класс информационно-справочных систем основан на гипертекстовых документах и мультимедиа. Наибольшее развитие такие информационные системы получили в Интернете.

Класс офисных информационных систем нацелен на перевод бумажных документов в электронный вид, автоматизацию делопроизводства и управление документооборотом.

3. Классификация по способу организации

По способу организации групповые и корпоративные информационные системы подразделяются на следующие классы (рис. 3):

- системы на основе архитектуры файл-сервер;
- системы на основе архитектуры клиент-сервер;
- системы на основе многоуровневой архитектуры;
- системы на основе Интернет/интранет-технологий.

Рис. 3. Деление информационных систем по способу организации.

В любой информационной системе можно выделить необходимые функциональные компоненты (табл. 1), которые помогают понять ограничения различных архитектур информационных систем. Рассмотрим более подробно особенности вариантов построения информационных приложений.

Таблица 1.1. Типовые функциональные компоненты информационной системы

Обозначение	Наименование	Характеристика
PL	Presentation Logic (логика представления)	Управляет взаимодействием между пользователем и ЭВМ. Обрабатывает действия пользователя при выборе команды в меню, щелчке на кнопке или выборе пункта в списке
BL	Business Logic (прикладная логика)	Набор правил для принятия решений, вычислений и операций, которые должно выполнить приложение
DL	Data Logic (логика управления данными)	Операции с базой данных (реализуемые SQL-операторами), которые нужно выполнить для реализации прикладной логики управления данными
DS	Data Services (операции с базой данных)	Действия СУБД, реализующие логику управления данными, такие как манипулирование данными, определение данных, фиксация или откат транзакций и т. п. СУБД обычно компилирует SQL-предложения
FS	File Services (файловые операции)	Дисковые операции чтения и записи данных для СУБД и других компонентов. Обычно являются функциями операционной системы (ОС)

2. Состав и функции средств актуализации БД, средств обработки БД в интересах пользователей, средств администрирования БД.

К основным функциям ИС относятся функции сбора и регистрации информационных ресурсов, их хранение, обработка, актуализация, а так же обработка запросов пользователя.

Сбор и регистрация обеспечивает фиксирование информации о состоянии предметной области. Работы выполняется как до основного программно-аппаратного комплекса, так в его среде. Реализация функций зависит от источника информации, в качестве которого могут выступать бумажные носители, электронные, автоматизированные технические системы.

Сбор и регистрация могут осуществляться:

- путем измерений (наблюдений) фактов в реальном мире и ввода данных в систему с помощью клавиатуры или каких-либо манипуляторов;
- полуавтоматически путем ввода в компьютер с некоторых носителей и в случае необходимости их перекодировать (например, при использовании текстов на бумажных носителях или аналоговых аудизаписей);
- автоматический с помощью различного рода датчиков или обмена данными с другими автоматизированными системами.

С этими функциями связана необходимость обеспечения контроля, сжатие, конвертирование информации.

Обеспечение контроля информации – необходимая стадия предварительной обработки данных и подготовки их загрузки в систему, особенно в случаях, когда используются несколько источников данных. Обычно она включает процедуры фильтрации данных, верификации, обеспечение логической целостности, устранение несогласованности, избыточности и различных ошибок, восполнения пропусков, а также другие процедуры направленные на улучшение качества информации.

В результате фильтрации производится отбор нужных данных из множества имеющихся в распоряжении. Верификации призвана обеспечивать достоверность и логическую целостность информации. При выполнении данной функции устанавливается, адекватна ли или информация предметной области.

На разных операциях могут применяться различные методы контроля, существуют методы, применимые ко многим операциям, наиболее применимые:

- подсчета контрольных сумм;
- повторное выполнение операций другим оператором с дублированием действий и последующим их сличением;
- контроль набора на клавиатуре;
- контроль информации в соответствие с ее свойствами, структурой и на соответствие значениям.

Способами реализации могут быть:

- ручной (без использования технических средств);
- визуальный (с использованием технических средств и без них);
- аппаратный (технический);
- программный;
- организационные мероприятия.

Выбор конкретных обеспечения верификации зависит от характера, качества, источников данных, видов ограничения целостности.

В некоторых ИС информация хранится в сжатом виде. Сжатие информации минимизирует потребность во внешней памяти, нужной для хранения, а также снижает затраты на передачу данных по каналам связи.

Конвертирование данных при вводе в систему используется для преобразования одного формата данных в другой, допускающий автоматизированный импорт их в ИС. Конвертирование данных необходимо в случаях, когда источником данных является некоторая другая система. Для конвертирования используются специальные программы конверторы.

Хранение и накопление информации вызвано необходимостью многократного использования одни и те же данные при решении задач. Для хранения и поиска информации используются технологии баз данных.

Актуализация информационных ресурсов. Для того, чтобы информация была практически полезной, необходимо своевременно и адекватно отображать в ней изменения состояния предметной области. Актуализация информации в реляционных СУБД сводится к включению и/или удалению строк в таблицах баз данных, обновлению значений некоторых реквизитов. В случаях изменения структуры предметной области системы, актуализация информации заключается в изменении схемы базы данных – добавлении или удалении существующих столбцов таблиц, в создании новых таблиц и удалении существующих таблиц.

В информационно-справочные системах актуализация информации осуществляется путем ввода в систему новых документов, реже удалением существующих.

Актуализация информации в ИС производится дискретно, через определенные интервалы времени. Актуализация информации, т.о., обеспечивается с некоторым отставание во времени. Это отставание в различных ИС изменяется в широком диапазоне и зависит от назначения системы и особенностей ее предметной области. В информационных системах управления сложными техническими объектами, например в системе управления

космическими полетами, временной лаг измеряется в миллисекундах. В корпоративных ИС может составлять от нескольких минут до нескольких часов.

Для того чтобы ИС соответствовала своему назначению необходимо соблюдать установленный для нее регламент актуализации.

Предоставление информационных ресурсов пользователю. Все выше описанные операции необходимы для удовлетворения информационных потребностей пользователей.

Существует две технологии предоставления информации пользователю: pull-технология и/ или push-технология.

В случае pull-технологии – инициатором предоставления информации выступает пользователь, а push-технология сама система, в соответствие с регламентом и для определенного круга пользователей.

Для предоставления информации по pull-технологии в ИС предусматриваются пользовательские интерфейсы. Пользовательские интерфейсы – средства взаимодействия пользователя с системой.

При этом пользователь может влиять на последовательность применения тех или иных технологий. С точки зрения влияния пользователя на последовательность операций в процессе функционирования ИС, интерфейсы могут быть разделены на пакетные и диалоговые.

Экономические задачи, решаемые в пакетном режиме, характеризуются следующими свойствами:

- алгоритм решения задачи формализован, процесс ее решения не требует вмешательства человека;
- имеется большой объем входных и выходных данных, значительная часть которых хранится на магнитных дисках;
- расчет выполняется для большинства записей входных файлов;
- большое время решения задачи обусловлено большими объемами данных;
- регламентность, т.е. задачи решаются с заданной периодичностью.

Диалоговый режим не является альтернативой пакетному режиму, а его развитием. Если применение пакетного режима позволяет уменьшить вмешательство пользователя в процесс решения задачи, то диалоговый режим предполагает отсутствие жестко закрепленной последовательности операций обработки данных.

Примером push-технологии может служить рассылка информации среди пользователей Интернет.

Экономическая информационная система по своему составу напоминает предприятие по переработке данных и производству выходной информации. Методы и способы реализации функции ИС (сбора, накопления, хранения, поиска и обработки информации на основе применения средств вычислительной техники) называются информационной технологией.

Информационные технологии должны быть выстроены в последовательность действий, позволяющую из исходной информации получить результат с заданной достоверностью и безопасностью.

Упорядоченная последовательность взаимосвязанных действий, выполняющихся с момента возникновения информации до получения результата, называется технологическим процессом.

Понятие информационной технологии, таким образом, неотделимо от той специфической среды, в которой она реализована, т.е. от технической и программной Среды.

Администрирование базы данных – это функция управления базой данных (БД). Лицо ответственное за администрирование БД называется “Администратор базы данных” (АБД) или “Database Administrator” (DBA).

Функция “администрирования данных” стала активно рассматриваться и определяться как вполне самостоятельная с конца 60-х годов. Практическое значение это имело для предприятий, использующих вычислительную технику в системах

информационного обеспечения для своей ежедневной деятельности. Специализация этой функции с течением времени совершенствовалась, но качественные изменения в этой области стали происходить с началом использования так называемых интегрированных баз данных. Одна такая база данных могла использоваться для решения многих задач.

Таким образом, сформировалось определение БД как общего информационного ресурса предприятия, которое должно находиться всегда в работоспособном состоянии. И как для каждого общего ресурса значительной важности, БД стала требовать отдельного управления. Во многих случаях это было необходимо для обеспечения её повседневной эксплуатации, её развития в соответствии с растущими потребностями предприятия. К тому же БД и технология её разработки постоянно совершенствовались и уже требовались специальные знания высокого уровня для довольно сложного объекта, которым стала база данных. Отсюда функция управления базой данных и получила название “Администрирование базы данных”, а лицо ею управляющее стали называть “Администратор баз данных”.

Администратор базы данных (DBA)

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

АБД имеет код специальности по общероссийскому классификатору профессий рабочих, должностей служащих и тарифных разрядов (ОКПДТР) — 40064 и код 2139 по Общероссийскому классификатору занятий (ОКЗ). Код 2139 ОКЗ расшифровывается следующим образом: 2 - СПЕЦИАЛИСТЫ ВЫСШЕГО УРОВНЯ КВАЛИФИКАЦИИ, 21 - Специалисты в области естественных* и инженерных наук, 213 - Специалисты по компьютерам, 2139 - Специалисты по компьютерам, не вошедшие в другие группы.

Классические подходы к наполнению содержанием понятия "АБД" стали формироваться после издания рабочего отчета группы по базам данных Американского Национального Института Стандартов ANSI/X3/SPARC в 1975 года. В этом отчете была описана трехуровневая архитектура СУБД, в которой выделялся уровень внешних схем данных, уровень концептуальной схемы данных и уровень схемы физического хранения данных. В соответствии с этой архитектурой определялись три роли АБД: администратор концептуальной схемы, администратор внешних схем и администратор хранения данных. Эти роли в случае очень маленькой системы мог играть один человек, в большой системе для выполнения каждой роли могла назначаться группа людей. Каждой роли соответствовал набор функций, а все эти функции вместе составляли функции АБД.

В 1980 - 1981 г. в американской литературе стало принятым включать в функции АБД:

- организационное и техническое планирование БД,
- проектирование БД,
- обеспечение поддержки разработок прикладных программ,
- управление эксплуатацией БД.

В нашей стране в это же время первое определение АБД в ГОСТ-ах задало слишком узкий состав функций АБД:

- подготовка вычислительного комплекса к установке СУБД, участие в установке и приемке СУБД и самой БД с комплексом прикладных программ
- управление эксплуатацией БД
- подготовка словарей и другой НСИ - нормативно-справочной информации - к моменту начала испытания БД
- Предполагалось, что функции АБД будут ориентированы только на эксплуатацию БД, а её разработка будет вестись силами специализированной организации.

К середине 90-х годов сложились еще не завершенные, но уже достаточно устойчивые и полные методологии разработки систем с базами данных. Основная работа по планированию информационных потребностей предприятия, проектированию концептуальной и логической схемы БД, внешних схем, используемых в отдельных процессах обработки информации, ложится теперь на группу проектирования Автоматизированной Системы (АС). Становиться и более определённым объем функций АБД. Это обеспечение надежной и эффективной работы пользователей и программ с БД, поддержка разработчиков в их доступе к БД и средствам разработки.

Задачи АБД могут незначительно отличаться в зависимости от вида применяемой СУБД

Среди АБД нет строгого документального разграничения по типам.

Как таковой официальной версии должностной инструкции администратора базы данных не существует. Имеется несколько документов, различающихся в основном оформлением и содержанием некоторых пунктов. Естественно, ни о какой подробной расшифровке задач администратора базы данных в этих документах речи не идёт. Должностная инструкция — это, прежде всего документ, регламентирующий производственные полномочия и обязанности работника. И хотя в некоторых организациях изменяют текст должностной инструкции в соответствии с условиями специфики работы, не стоит ожидать в ней прямых указаний на то, что надо делать администратору. В большинстве случаев такие детальные директивы работы регламентируются другими документами (например, инструкция по резервному копированию, инструкция по обеспечению информационной безопасности при работе с базами данных).

3. Технологии файл-сервер и клиент-сервер.

Архитектура файл‑сервер.

В архитектуре файл‑сервер сетевое разделение компонентов диалога PS и PL отсутствует, а компьютер используется для функций отображения, что облегчает построение графического интерфейса. Файл‑сервер только извлекает данные из файлов, так что дополнительные пользователи и приложения лишь незначительно увеличивают нагрузку на центральный процессор. Каждый новый клиент добавляет вычислительную мощность к сети.

Объектами разработки в файл‑серверном приложении являются компоненты приложения, определяющие логику диалога PL, а также логику обработки BL и управления данными DL. Разработанное приложение реализуется либо в виде законченного загрузочного модуля, либо в виде специального кода для интерпретации.

Однако такая архитектура имеет существенный недостаток: при выполнении некоторых запросов к базе данных клиенту могут передаваться большие объемы данных, загружая сеть и приводя к непредсказуемости времени реакции. Значительный сетевой трафик особенно сказывается при организации удаленного доступа к базам данных на файл‑сервере через низкоскоростные каналы связи. Одним из вариантов устранения данного недостатка является удаленное управление файл‑серверным приложением в сети. При этом в локальной сети размещается сервер приложений, совмещенный с телекоммуникационным сервером (обычно называемым сервером доступа), в среде которого выполняются обычные файл‑серверные приложения. Особенность состоит в том, что диалоговый ввод‑вывод поступает от удаленных клиентов через телекоммуникации. Приложения не должны быть слишком сложными, иначе велика вероятность перегрузки сервера, или же нужна очень мощная платформа для сервера приложений.

Одним из традиционных средств, на основе которых создаются файл‑серверные системы, являются локальные СУБД. Однако такие системы, как правило, не отвечают требованиям обеспечения целостности данных (в частности, они не поддерживают транзакции). Поэтому при их использовании задача обеспечения целостности данных возлагается на программы клиентов, что приводит к усложнению клиентских приложений.

Тем не менее, эти инструменты привлекают своей простотой, удобством применения и доступностью. Поэтому файл-серверные информационные системы до сих пор представляют интерес для малых рабочих групп и, более того, нередко используются в качестве информационных систем масштаба предприятия.

Архитектура клиент-сервер.

Архитектура клиент-сервер предназначена для разрешения проблем файл-серверных приложений путем разделения компонентов приложения и размещения их там, где они будут функционировать наиболее эффективно. Особенностью архитектуры клиент-сервер является наличие выделенных серверов баз данных, понимающих запросы на языке структурированных запросов (Structured Query Language, SQL) и выполняющих поиск, сортировку и агрегирование информации.

Отличительная черта серверов БД – наличие справочника данных, в котором записана структура БД, ограничения целостности данных, форматы и даже серверные процедуры обработки данных по вызову или по событиям в программе. Объектами разработки в таких приложениях помимо диалога и логики обработки являются, прежде всего, реляционная модель данных и связанный с ней набор SQL-операторов для типовых запросов к базе данных.

Большинство конфигураций клиент-сервер использует двухуровневую модель, в которой клиент обращается к услугам сервера. Предполагается, что диалоговые компоненты PS и PL размещаются на клиенте, что позволяет реализовать графический интерфейс. Компоненты управления данными DS и FS размещаются на сервере, а диалог (PS, PL) и логика (BL, DL) – на клиенте. В двухуровневом определении архитектуры клиент-сервер используется именно этот вариант: приложение работает на клиенте, СУБД – на сервере (рис. 4).

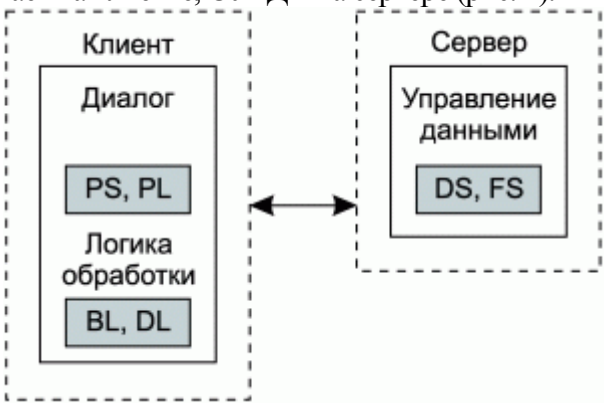


Рис. 4. Классический вариант клиент-серверной системы.

Поскольку эта схема предъявляет наименьшие требования к серверу, она обладает наилучшей масштабируемостью. Однако сложные приложения, активно взаимодействующие с БД, могут жестко загрузить как клиента, так и сеть. Результаты SQL-запроса должны вернуться клиенту для обработки, потому что там реализована логика принятия решения. Такая схема приводит к дополнительному усложнению администрирования приложений, разбросанных по различным клиентским узлам.

Для сокращения нагрузки на сеть и упрощения администрирования приложений компонент BL можно разместить на сервере. При этом вся логика принятия решений оформляется в виде хранимых процедур и выполняется на сервере БД.

Хранимая процедура – процедура с SQL-операторами для доступа к БД, вызываемая по имени с передачей требуемых параметров и выполняемая на сервере БД. Хранимые процедуры могут компилироваться, что повышает скорость их выполнения и сокращает нагрузку на сервер.

Хранимые процедуры улучшают целостность приложений и БД, гарантируют актуальность коллективных операций и вычислений. Улучшается сопровождение таких процедур, а также безопасность (нет прямого доступа к данным).

2. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

2.1 Лабораторная работа №1, 2, 3, 4 (8 часов).

Тема: «Постановка и решение задачи линейного программирования в MS Excel»

2.1.1 Цель работы: изучить и проанализировать решение задачи линейного программирования в MS Excel

2.1.2 Задачи работы:

1. Изучение постановки задачи линейного программирования
2. Изучение решения задачи линейного программирования
3. Изучение решения задачи линейного программирования в MS Excel

2.1.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. OpenOffice
2. Microsoft Office Excel

2.1.4 Описание (ход) работы:

1. Формализация прикладных экономических задач в виде основной задачи линейного программирования.

Постановка задачи предполагает четкую экономическую формулировку, включающую цель решения, установление планового периода, выяснение известных параметров объекта и тех, количественное значение которых нужно определить, их производственно-экономических связей, а также множества факторов и условий, отражающих моделируемый процесс.

Модель экономической задачи состоит из трех частей.

I. Целевая функция (критерий оптимальности), описывающая конечную цель, преследуемую при решении задачи.

II. Система ограничений, которая включает основные и дополнительные ограничения. Основные ограничения описывают расход производственных ресурсов (консервативная часть модели), дополнительные – имеют различный характер, являются изменяемой частью модели, с помощью которой отражаются особенности моделируемой задачи.

III. Условие неотрицательности переменных величин.

Остановимся подробнее на каждой части модели.

Так, цель решения задачи выражается количественно конкретным показателем, называемым критерием оптимальности. Он должен соответствовать экономической сущности решаемой задачи.

Существует достаточно большое количество локальных критериев оптимизации, используемых как в промышленном производстве, так и в сельском хозяйстве:

– максимум производства валовой продукции в стоимостном выражении;

– максимум валового дохода, представляющего разницу между валовой продукцией в стоимостном выражении и суммой материальных затрат на ее производство;

– максимум чистого дохода, измеряемого разницей между стоимостью валовой продукции и суммой издержек производства;

– максимум прибыли, измеряемой разницей между суммой денежных поступлений от реализации продукции и ее полной себестоимостью;

– минимум производственных затрат на заданный план производства продукции, исчисляемых по формуле:

$$C = S + a\,k,$$

где S – текущие производственные затраты;

k – удельные капиталовложения;

a – норма эффективности капиталовложений;

– максимум приведенной прибыли, измеряемой разницей между валовой выручкой за реализованную продукцию и приведенными затратами на ее производство;

– максимум денежных поступлений от реализации продукции;

– минимум производственных затрат на заданный план производства продукции.

В постановке задачи должно быть четко определено, что является неизвестным, какие переменные величины и их численные значения необходимо найти в процессе решения.

Перечень переменных величин должен отражать характер, основное содержание моделируемого экономического процесса.

Количество переменных зависит от выбора планового периода (долгосрочный, среднесрочный, текущий), который оказывает существенное влияние на степень их детализации. Чем ближе период, на который составляется модель, тем больше детализация переменных. При планировании на более отдаленную перспективу (пятилетний план, план организационно-хозяйственного устройства на перспективу) необходимости в столь подробной детализации переменных нет.

Кроме того, количество переменных зависит и от того, насколько подробно в модели должны быть представлены следующие признаки: вид продукции; направление ее использования; способы, каналы и сроки производства и реализации продукции.

По экономической роли в моделируемом процессе все переменные классифицируются на *основные* и *вспомогательные*.

К *основным* переменным можно отнести:

– объемы производства продукции (бытовой техники, строительных и отделочных материалов, мебели, и т.д.);

– количество сырья, используемого для промышленного производства;

– количество промышленного оборудования;

– поголовье сельскохозяйственных животных;

– площади посева сельскохозяйственных угодий;

– количество сельскохозяйственной техники;

– количество минеральных удобрений и пр.

Вспомогательные переменные привлекают для облегчение математической формулировки условий, определения расчетных величин (объемов производства, показателей эффективности производства и т.д.).

При математической реализации задач для преобразования неравенств в равенства вводятся дополнительные переменные, которые используются при анализе промежуточных решений и оптимального варианта.

Для каждой переменной устанавливают конкретную единицу измерения (шт., га, ц., чел.-ч., и т.д.) При этом руководствуются следующими требованиями:

1) целесообразно выбирать одинаковые единицы измерения по однотипным группам переменных;

2) единицы измерения не должны затруднять анализ оптимального решения и вызывать дополнительные расчеты;

3) технико-экономические коэффициенты нецелесообразно представлять слишком большими или слишком малыми числами.

После установления состава переменных определяют систему ограничений модели, отражающих условия реализации задачи. Ограничения, представленные в виде линейных неравенств и уравнений, отражают организационно-экономические и технологические условия и требования, которые характеризуют данное производство.

Ограничение записываются тремя типами линейных соотношений: меньше или равно (≤), больше или равно (≥) и равно (=). По своей роли в модели они подразделяются на основные, дополнительные и вспомогательные.

Основные ограничения выражают главные, наиболее существенные условия задачи. Они накладываются на все или большинство переменных моделей. К основным относятся ограничения по использованию производственных ресурсов.

Дополнительные ограничения накладываются на небольшое количество переменных величин или отдельные переменные. Обычно они формулируются в виде неравенств.

Вспомогательные ограничения вводят для облегчения разработки числовой модели, обеспечение правильной формулировки экономических требований. Самостоятельного экономического значения не имеют. С помощью вспомогательных ограничений могут быть записаны условия пропорциональной связи между переменными или их группами.

Размеренность величин каждого ограничения определяется размеренностью его правой части. Если она, например, означает запас ресурсов труда в человеко-часах, то в левой части ограничения показатели по использованию трудовых ресурсов по всем видам деятельности также выражаются в человеко-часах.

В зависимости от задачи и объекта, по которому эта задача должна быть построена, необходимо определить *характер и объем информации, источники ее сбора и методы обработки*.

Источниками информации служат годовые отчеты, производственно-финансовые и перспективные планы, планы организационно-хозяйственного устройства, данные первичного учета, технологические карты производства различных видов продукции, а также различные нормативные справочники.

Целью переработки исходной информации являются разработка и обоснование системы технико-экономических характеристик объекта или процесса. Для любой модели эти характеристики формируются в виде технико-экономических коэффициентов a_{ij}, коэффициентов целевой функции c_j и констант или объемных показателей ресурсов или продуктов b_i.

Технико-экономические коэффициенты представляют собой основную часть входной информации, которая поступает в модель как в преобразованном, так и не в преобразованном виде. Коэффициенты можно подразделить на группы: удельные нормативы затрат или выхода продукции, коэффициенты пропорциональности.

Удельные нормативы затрат или выхода продукции представляют собой технико-экономическую характеристику видов (способов) деятельности. По экономическому содержанию выделяют коэффициенты, характеризующие затраты i-го ресурса на единицу j-го вида деятельности – a_{ij} и коэффициенты выхода v_{ij}.

Удельные коэффициенты затрат и выхода рассчитывают на основе нормативных справочников, технологических карт с использованием методов математической статистики и другими способами. От их достоверности зависит результат решения задачи. Единицы измерения этих величин определяются отношением единицы измерения b_i, к единице измерения x_j.

Коэффициенты пропорциональности (W_{ij}) – это коэффициенты при переменных в тех ограничениях, которые предусматривают определенные пропорции между зависимыми переменными.

Экономическое содержание коэффициентов в целевой функции (c_j) определяется характером критерия оптимальности. Числовое значение критерия оптимальности чаще всего исчисляются как сумма произведений коэффициентов целевой функции и значений переменных, то есть $\sum c_i x_j$.

Большое число экономических задач сводится к линейным математическим моделям. Традиционно оптимизационные линейные математические модели называются модели линейного программирования. Под линейным программированием понимается линейное планирование, т.е. получение оптимального плана решения в задачах, имеющих линейную структуру.

Основная задача линейного программирования звучит следующим образом: пусть некоторые предприятие имеет m видов производственных ресурсов, порядковый номер ресурсов – i , т.е. i = 1,2,3,…, m; наличие каждого вида ресурсов известно и обозначается b_i;

предположим, что предприятие может производить n видов продукции, порядковый номер продукции – j, т.е. $j = 1,2,3,\ldots, n$; необходимо определить какое количество единиц продукции каждого вида надо производить (x_j), чтобы получить максимум этой продукции в стоимостном выражении, если известно, что затраты на производство единицы продукции каждого вида ресурса a_{ij} единиц и цена реализации – c_j.

Общий вид экономико-математической модели:

I. $Z = c_1x_1 + c_2x_2 + \ldots + c_nx_n \rightarrow \max$.

II. $a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \le b_1$,

$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \le b_2$,

……

$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \le b_m$.

III. $x_1 \ge 0, x_2 \ge 0, \ldots, x_n \ge 0$.

Структурная форма записи экономико-математической модели:

I. $Z = \sum_{j=1}^{n} c_j x_j \rightarrow \max$.

II. $\sum_{j=1}^{n} a_{ij} x_j \ge b_i, i = 1,2,...,m.$

III. $x_j \ge 0, j = 1,2, \ldots n.$

Целевая функция в модели может стремиться как к max, так и к min, исходя из условия задачи.

2. Решение задач линейного программирования в MS Excel. Экономическая интерпретация результатов решения задач.

Негосударственный пенсионный фонд России «Галина» решил инвестировать свободные денежные средства в ценные бумаги разных компаний. На фондовой бирже интересы фонда представлены тремя инвесторами («Инвест-Компани»; «Рус-Инвест»; «Инвест-Гарант»). Они могут разместить имеющийся капитал в четырех компаниях (ОАО «Русь»; ОАО «Заря»; ОАО «Луч»; ОАО «Мир»). Доходность каждой ценной бумаги и ее стоимость представлены в таблице 1.

Необходимо найти максимально возможную прибыль негосударственного пенсионного фонда «Галина» от инвестирования в ценные бумаги.

Решение

1. Состав переменных:

x_1 – количество ценных бумаг ОАО «Русь»,

x_2 – количество ценных бумаг ОАО «Заря»,

x_3 – количество ценных бумаг ОАО «Луч»,

x_4 – количество ценных бумаг ОАО «Мир».

Таблица 1 – Исходные данные к задаче

Инвестор	Цена ценной бумаги				Ресурсы
	ОАО "Русь"	ОАО "Заря"	ОАО "Луч"	ОАО "Мир"	
"Инвест-Компани"	4	2	2	3	35
"Рус-Инвест"	1	1	2	3	30
"Инвест-Гарант"	3	1	2	1	40
Доходность	14	10	14	11	

2. Целевая функция. Критерий оптимальности – получение максимальной суммарной доходности пенсионного фонда, исходя из имеющихся денежных ресурсов. Тогда модель будет выглядеть следующим образом:

I. $Z = 14x_1 + 10x_2 + 14x_3 + 11x_4 \rightarrow \max.$

II. $4x_1 + 2x_2 + 2x_3 + 3x_4 \leq 35,$

$x_1 + x_2 + 2x_3 + 3x_4 \leq 30,$

$3x_1 + x_2 + 2x_3 + x_4 \leq 40.$

III. $x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0.$

Для решения поставленной задачи воспользуемся табличным редактором MS Excel.

В ячейку A1 запишем целевую функцию (рисунок 1). В ячейки B1:B3 запишем основные ограничения до знака неравенства.

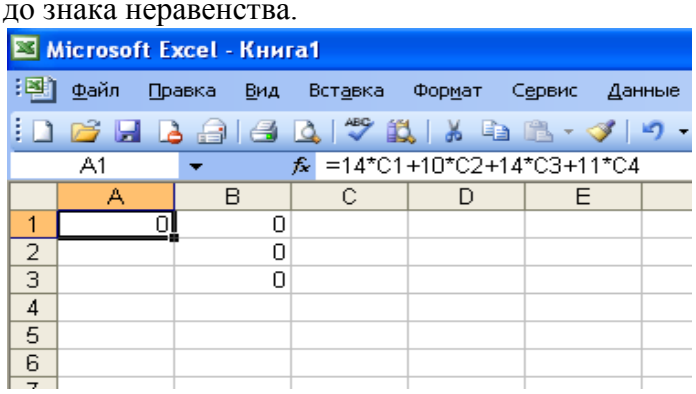


Рисунок 1 – Целевая функция

После того, как числовая модель записана, необходимо установить курсор в ячейку A1, в которой расположена целевая функция. Далее выбираем вкладку «Сервис» – «Поиск решения», при этом откроется диалоговое окно данной функции, представленное на рисунке 2.

Далее устанавливаем курсор в ячейку A1, выбираем вкладку «Сервис» - «Поиск решения» и в появившемся диалоговом окне устанавливаем параметры, которые представлены на рисунке 2.

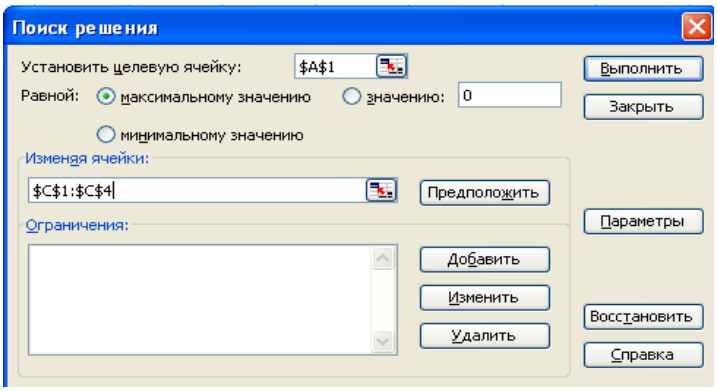


Рисунок 2 – Поиск решения

В открывшемся окне необходимо установить целевую ячейку, а поскольку у вас курсор стоял на ячейке A1, то значение целевой ячейки будет верным. В противном случае необходимо установить адрес целевой ячейки вручную.

Следующим шагом будет установление маркера в положение соответствующего критерия оптимальности: максимальное или минимальное значение. В окне «Изменяя ячейки» следует указать адреса ячеек, соответствующих переменным. Для этого необходимо выделить диапазон ячеек с C1 по C4.

В окне «Ограничения» следует активировать кнопку «Добавить». Откроется окно «Добавить ограничения», представленное на рисунке 3.

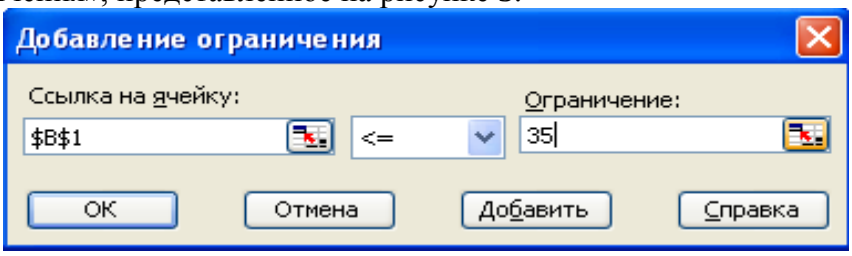


Рисунок 3 – Добавление ограничений

Для ввода первого ограничения в окне «Ссылка на ячейку» указываем адрес ячейки, где находится левая часть 1-го ограничения $B$1, затем выбираем знак ограничения «≤», а в поле «ограничение» – значение 35. Активируем клавишу «Добавить» и аналогично вводим оставшиеся ограничения. Затем вводим условие неотрицательности. Для этого в окне «Ссылка на ячейку» указываем диапазон ячеек, в которых находятся переменные ($C$1: $C$4). Аналогично вводится условие целочисленности переменных.

Указав все условия задачи, переходим к поиску максимальной доходности пенсионного фонда (рисунок 4).

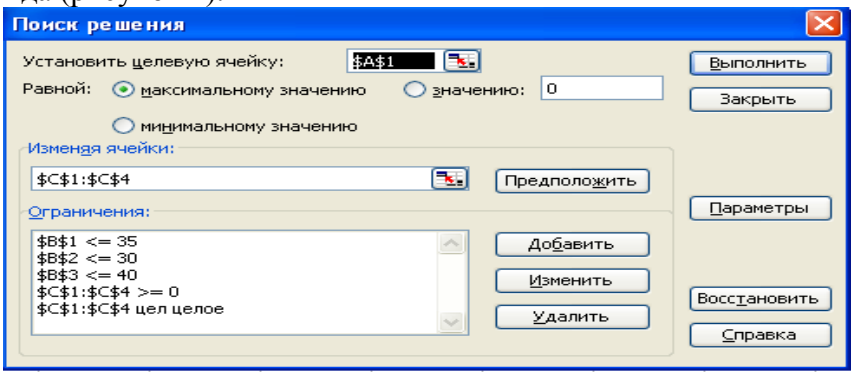


Рисунок 4 – Поиск максимальной доходности пенсионного фонда

Выбираем команду «Выполнить». На экране появится окно «Результаты поиска решения» (рисунок 5).

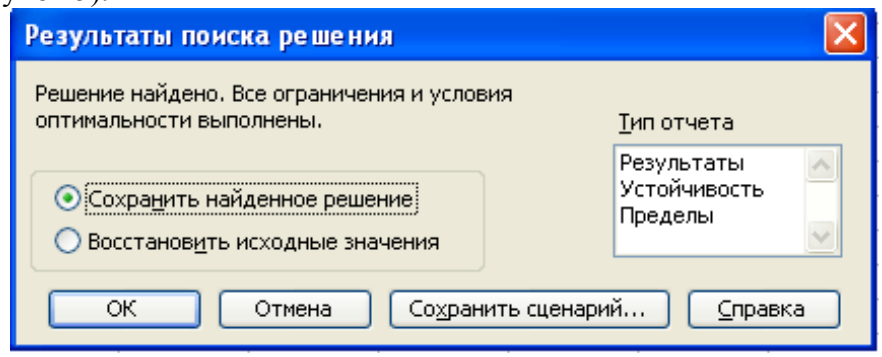
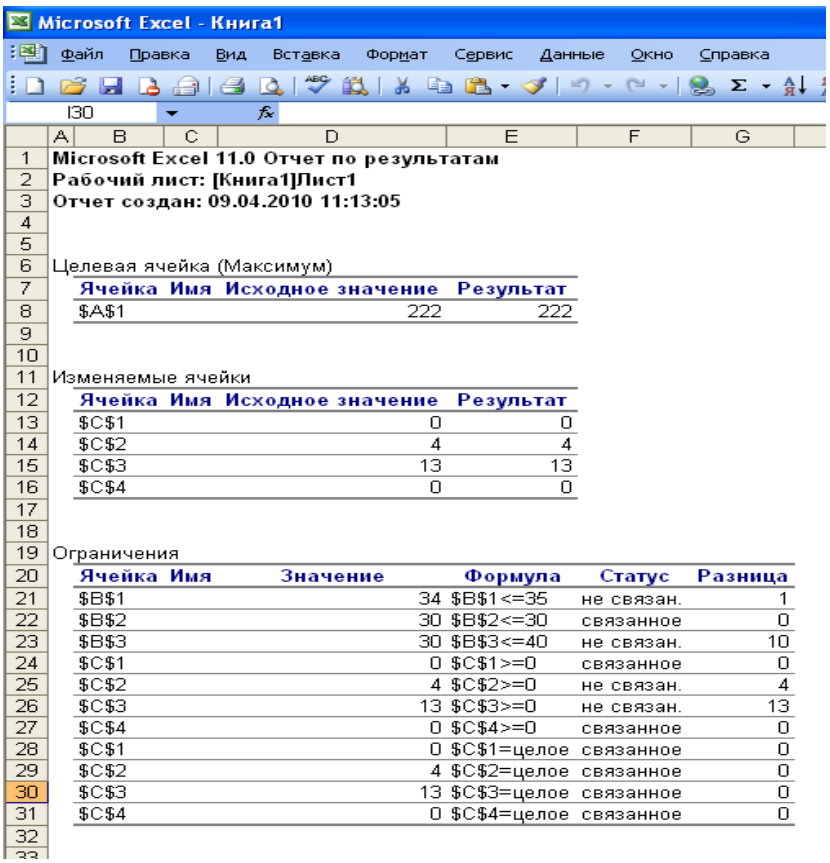


Рисунок 5 – Результаты поиска решения

Если модель составлена правильно и имеет решение, в открывшемся окне будет сообщение «Решение найдено. Все ограничения и условия оптимальности выполнены». Для просмотра результатов решения задачи в окне «Тип отчета» необходимо выбрать «Результаты» и нажать «ОК». В рабочем файле появится новый лист «Отчет по результатам», в котором представлено решение задачи (рисунок 6). Решение задачи окончено, можно распечатать результаты.



Microsoft Excel - Книга1

Файл Правка Вид Вставка Формат Сервис Данные Окно Справка

I30

Microsoft Excel 11.0 Отчет по результатам
Рабочий лист: [Книга1]Лист1
Отчет создан: 09.04.2010 11:13:05

Целевая ячейка (Максимум)

Ячейка	Имя	Исходное значение	Результат
$A$1		222	222

Изменяемые ячейки

Ячейка	Имя	Исходное значение	Результат
$C$1		0	0
$C$2		4	4
$C$3		13	13
$C$4		0	0

Ограничения

Ячейка	Имя	Значение	Формула	Статус	Разница
$B$1		34	$B$1<=35	не связан.	1
$B$2		30	$B$2<=30	связанное	0
$B$3		30	$B$3<=40	не связан.	10
$C$1		0	$C$1>=0	связанное	0
$C$2		4	$C$2>=0	не связан.	4
$C$3		13	$C$3>=0	не связан.	13
$C$4		0	$C$4>=0	связанное	0
$C$1		0	$C$1=целое	связанное	0
$C$2		4	$C$2=целое	связанное	0
$C$3		13	$C$3=целое	связанное	0
$C$4		0	$C$4=целое	связанное	0

Рисунок 6 – Отчет по результатам

Результат, полученный в ячейке A1, представляет значение целевой функции. Из приведенных данных мы видим, что максимально возможный доход от вложения в ценные бумаги равен 222 единицам. Ячейки C1:C4 указывают на количество приобретенных ценных бумаг. Для получения максимального дохода необходимо приобрести 4 акции компании ОАО «Заря» и 13 акций ОАО «Луч». После совершенных действий остаток ресурсов «Инвест-Компани» составит 1 единицу, а «Инвест-Гарант» – 10 единиц, а ресурсы «Рус-Инвест» будут исчерпаны полностью.

Примечания

Если поиск не может найти оптимальное решение, в диалоговом окне **Результат поиска решения** выводится одно из следующих сообщений.

Поиск не может улучшить текущее решение. Все ограничения выполнены.

В процессе поиска решения нельзя найти такой набор значений влияющих ячеек, который был бы лучше текущего решения. Приблизительное решение найдено, но либо дальнейшее уточнение невозможно, либо погрешность, заданная в диалоговом окне **Параметры поиска решения** слишком высока. Измените погрешность на меньшее число и запустите процедуру поиска решения снова.

Поиск остановлен (истекло заданное на поиск время).

Время, отпущенное на решение задачи, исчерпано, но достичь удовлетворительного решения не удалось. Чтобы при следующем запуске процедуры поиска решения не повторять выполненные вычисления, установите переключатель **Сохранить найденное решение** или **Сохранить сценарий.**

Поиск остановлен (достигнуто максимальное число интеграций).

Произведено разрешенное число интеграций, но достичь удовлетворительного решения не удалось. Увеличение числа итераций может помочь, однако следует рассмотреть результаты, чтобы понять причины остановки. Чтобы при следующем запуске процедуры поиска решения не повторять выполненные вычисления, установите переключатель **Сохранить найденное решение** или нажать кнопку **Сохранить сценарий.**

Значение целевой ячейки не сходятся.

Значение целевой ячейки неограниченно увеличивается (или уменьшается), даже если все ограничения соблюдены. Возможно следует в задаче снять одно ограничение или сразу несколько. Изучите процесс расхождения решения, проверьте ограничения и запустите задачу снова.

Поиск не может найти подходящего решения.

В процессе поиска решения нельзя сделать итерацию, которая удовлетворяла бы всем ограничениям при заданной точности. Вероятно, ограничения противоречивы. Исследуете лист на предмет возможных ошибок в формулах ошибок в формулах ограничений или в выборе ограничений.

Поиск остановлен по требованию пользователя.

Нажата кнопка **Стоп** в диалоговом окне **Текущее состояние поиска решения** после прерывания поиска решения в процессе выполнения итераций.

Условия для линейной модели не удовлетворяются.

Установлен флажок **Линейная модель,** однако итоговый пересчет порождает такие значения, которые не согласуются с линейной моделью. Это означает, что решение недействительно для данных формул листа. Чтобы проверить линейность задачи, установите флажок **Автоматическое масштабирование** и повторно запустите задачу. Если это сообщение опять появится на экране, снимите флажок **Линейная модель** и снова запустите задачу.

При поиске решения обнаружено ошибочное значение в целевой ячейке или в ячейке ограничения.

При пересчете значений ячеек обнаружена ошибка в одной формуле или в нескольких сразу. Найдите целевую ячейку или ячейку ограничения, порождающие ошибку, и измените их формулы так, чтобы они возвращали подходящее числовое значение.

Набранное неверное имя или формула в окне **Изменить ограничения**, либо в поле **Ограничения** было заданно целое или двоичное ограничение. Чтобы ограничить значение ячейки множеством целых чисел выберите оператора **целого** ограничения в списке условных операторов. Чтобы установить двоичное ограничение, выберите оператор для **двоичного** ограничения.

Задания

Задача 1

Кожгалантерейная фабрика выпускает три вида продукции: кожаные перчатки, ремни и сумочки. Согласно заключенным с магазинами договорам фабрика должна еженедельно поставлять не менее 70 пар перчаток, 30 ремней и 60 сумочек. Ресурсы на неделю следующие: 700 единиц труда, 580 единиц производственного оборудования, 600 единиц сырья, 540 единиц электроэнергии, расход которых на одну номенклатурную единицу продукции представлен в таблице 2.

Таблица 2 – Исходные данные к задаче 1

Ресурсы	Вид продукции		
	Перчатки	Ремни	Сумочки
Труд	3	3	4
Оборудование	2	3	4
Сырье	1	2	5
Электроэнергия	3	4	2

Цена перчаток равна 350 денежным единицам, ремней – 520 денежным единицам и сумочек – 700 денежным единицам.

Необходимо определить, сколько единиц каждого вида продукции надо выпускать, чтобы общая стоимость выпускаемой продукции была максимальной.

Задача 2

Для производства трех видов блокнотов бумажная фабрика использует два вида ресурса: натуральную кожу и бумагу. Нормы затрат ресурсов на блокнот, прибыль от реализации одного изделия и общее количество имеющихся ресурсов каждого вида приведены в таблице 3.

Таблица 3 –Исходные данные к задаче 2

Ресурсы	Нормы затрат ресурсов на одно изделие			Общее количество ресурсов
	Блокнот 1	Блокнот 2	Блокнот 3	
Натуральная кожа, $м^2$	0,25	0,15	0,1	30
Бумага, $м^2$	5	3	2	400
Трудоёмкость, чел.-ч.	1,5	1	0,75	400
Прибыль от реализации одного изделия, руб.	250	200	150	

Определить, сколько блокнотов каждого вида фабрике следует изготовить, чтобы прибыль от их реализации была максимальной.

Задача 3

Торговая фирма для продажи товаров 3-х видов использует ресурсы: время и площадь торговых залов. Затраты ресурсов на продажу одной партии товаров каждого вида приведены в таблице 4.

Таблица 4 – Исходные данные к задаче 3

Ресурсы	Вид товара			Объём ресурсов
	1	2	3	
Время, чел.-ч.	0,5	0,7	0,6	370
Площадь, $м^2$	0,1	0,3	0,2	90
Прибыль от реализации одной партии товара, у.е.	5	8	6	-

Прибыль, получаемая от реализации одной партии товара 1-го вида – 5 условных единиц, 2-го вида – 8 условных единиц, 3-го вида – 6 условных единиц. Определить оптимальную структуру товарооборота, обеспечивающую фирме максимальную прибыль.

Задача 4

На основе имеющихся данных определить оптимальную структуру активов и пассивов банка для максимизации текущей прибыли. Производственные ресурсы:

1) трудовые ресурсы - 15000 чел. часов;
2) офисные площади - 10000 клиентов в год;

Собственный капитал – 100 млн. руб. Норматив М1 (max Активы/ СК) – 10. Норма обязательных отчислений в резерв – 10% от привлеченных средств.

Годовые процентные ставки:

по кредитам, выданным:

- физ. лицам – 16%,
- юр. лицам – 12%,

– купонный доход по облигациям – 8%.

по депозитам привлеченным:

– физ. лиц – 4%,
– юр. лиц – 6%,
– по межбанковскому кредиту – 8%.

Удельные затраты производственных ресурсов на 1 млн. привлеченных и размещенных средств представлены в таблице 5.

Таблица 5 – Исходные данные к задаче 4

Показатели	Затраты труда, чел. дней	Использование офисных площадей, клиентов
Кредиты, выданные физ. лицам	40	40
Кредитам, выданные юр. лицам	20	5
Облигации	1	0
Депозиты привлеченные, физ. лиц	40	50
Депозиты привлеченные, юр. лиц	20	10
Межбанковский кредит	1	0

Политикой банка установлено, что он должен иметь не менее 8000 клиентов – физ. лиц. Существуют ограничения по привлечению межбанковского кредита – не более 200 млн. рублей.

3. Контрольная работа.

2.2 Лабораторная работа № 5, 6, 7 (6 часов).

Тема: «Формализация экономических задач и их решение на основе модели транспортной задачи. Использование для решения MS Excel»

2.2.1 Цель работы: изучить способы формализации экономических задач и их решение на основе модели транспортной задачи. Использовать для решения задач программный продукт MS Excel

2.2.2 Задачи работы:

1. Изучение способов формализации экономических задач

2. Изучение способов решения экономических задач на основе модели транспортной задачи

3. Использование программного продукта MS Excel при решение данных задач

2.2.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. OpenOffice

2. Microsoft Office Excel

2.2.4 Описание (ход) работы:

1. Стандартная модель транспортной задачи.

Среди проблем, для исследования которых успешно применяется линейное программирование, важное значение имеет так называемая транспортная задача.

Общая постановка этой задачи применительно к экономической проблеме экономии издержек производства формулируется так: имеется несколько пунктов назначения

(предприятий, потребителей, a_j, $j=\overline{1,n}$); требуется перевезти некоторое количество однородного товара (x_{ij}) из различных пунктов отправления (поставщики, b_i, $i=\overline{1,m}$) в несколько пунктов назначения; каждый из поставщиков может выделить только определенное количество единиц товара и каждому потребителю требуется также определенное количество единиц этого товара; известны расстояния или стоимости перевозки единицы товара от каждого поставщика к каждому потребителю (c_{ij}). Задача состоит в том, чтобы найти такие маршруты перевозок, из всех возможных связей поставщиков и потребителей, при которых общие транспортные расходы были бы минимальными.

Общий вид экономико-математической модели:

I. $Z = c_{11}x_{11} + c_{12}x_{12} + \ldots + c_{1n}x_{1n} + c_{21}x_{21} + c_{22}x_{22} + \ldots + c_{2n}x_{2n} + \ldots + c_{m1}x_{m1} + + c_{m2}x_{m2} + \ldots + c_{mn}x_{mn} \to \min.$

II. 1) $x_{11} + x_{12} + \ldots + x_{1n} = b_1;$
$x_{21} + x_{22} + \ldots + x_{2n} = b_2;$
$\ldots$
$x_{m1} + x_{m2} + \ldots + x_{mn} = b_m.$

2) $x_{11} + x_{21} + \ldots + x_{m1} = a_1;$
$x_{12} + x_{22} + \ldots + x_{m2} = a_2;$
$\ldots$
$x_{1n} + x_{2n} + \ldots + x_{mn} = a_n.$

III. $x_{11} \geq 0,\ x_{12} \geq 0,\ \ldots,\ x_{mn} \geq 0.$

Структурная форма записи экономико-математической модели:

I. $Z = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \to \min.$

II. 1) $\sum_{j=1}^{n} x_{ij} = b_i, \quad i=\overline{1,m}.$

2) $\sum_{i=1}^{n} x_{ij} = a_j, \quad j=\overline{1,n}.$

III. $x_{ij} \geq 0, \quad i=\overline{1,m},\ j=\overline{1,n}.$

Транспортная задача может быть решена с помощью одного из распределительных методов. С помощью алгоритма, разработанного для решения транспортной задачи, решаются многие экономические задачи, не имеющие характера перевозок, но условия, которых укладываются в модель транспортной задачи (распределение посевных площадей, составление различных смесей, размещение предприятий, раскрой материала и т.д.).

Этапы решения транспортной задачи:

1) проверка сбалансированности задачи;
2) определение переменных;
3) построение сбалансированной транспортной матрицы;
4) задание целевой функции;
5) задание ограничений;
6) решение задачи в Excel;
7) анализ результатов решения задачи:

В общей постановке в финансово-экономической интерпретации транспортная задача выглядит следующим образом.

Имеется m коммерческих банков с денежными средствами b_i в каждом. Имеется n держателей банковских карт с потребностью в денежных средствах a_j. Стоимость перевода денежных средств от m-го коммерческого банка до n-го держателя банковских карт равна c_{ij}.

Для модели транспортной задачи примем следующие обозначения:

a_j – потребность в денежных средствах держателей банковских карт (j = 1,2,3, …, n);

b_i – наличие денежных средств у коммерческих банков (i = 1,2,3, …, m);

c_{ij} – стоимость перевода денежных средств от i-го коммерческого банка к j-му держателю банковских карт.

Матрица c_{ij} называется матрицей тарифов (издержек).

Планом транспортной задачи называется матрица $X = (x_{ij})$ где каждое число x_{ij} обозначает количество денежных средств, которые необходимо перечислить из i-го банка к j-му держателю. Матрица X называется еще матрицей перевозок. Чаще всего матрицы тарифов и перевозок совмещают в одну двойную матрицу (таблица 1).

Если наличие денежных средств и потребности равны между собой, то задача является закрытой (сбалансированной):

$$\sum_{i=1}^{m} b_i = \sum_{j=1}^{n} a_j . \quad (1)$$

Если наличие денежных средств и потребностей не совпадают между собой, задача является открытой (несбалансированной):

$$\sum_{i=1}^{m} b_i \neq \sum_{j=1}^{n} a_j . \quad (2)$$

Таблица 1 – Общий вид транспортной матрицы

Коммерческие банки	Держатели банковских карт					Наличие денежных средств
	1	2	3	…	n	
1	c_{11} x_{11}	c_{12} x_{12}	c_{13} x_{13}	…	c_{1n} x_{1n}	b_1
2	c_{21} x_{21}	c_{22} x_{22}	c_{23} x_{23}	…	c_{2n} x_{2n}	b_2
3	c_{31} x_{31}	c_{32} x_{32}	c_{33} x_{33}	…	c_{3n} x_{3n}	b_3
…	…	…	…	…	...	...
m	c_{n1} x_{m1}	c_{n2} x_{m2}	c_{n3} x_{m3}	…	c_{mn} x_{mn}	b_m
Потребность в денежных средствах	a_1	a_2	a_3	…	a_n	$\sum_{j=1}^{n} a_j = \sum_{i=1}^{m} b_i$

В случае, когда *суммарные запасы денежных средств коммерческих банков превышают суммарные потребности в денежных средствах держателей банковских карт*, необходим дополнительный фиктивный держатель банковской карты, который будет формально использовать существующий излишек денежных средств, то есть:

$$a_{ф} = \sum_{i=1}^{m} b_i - \sum_{j=1}^{n} a_j . \quad (3)$$

В случае, когда *суммарная потребность в денежных средствах держателей банковских карт превышают суммарные запасы денежных средств коммерческих банков,*

необходим фиктивный коммерческий банк, формально восполняющий существующий недостаток денежных средств держателей банковских карт:

$$b_{\phi} = \sum_{j=1}^{n} a_j - \sum_{i=1}^{m} b_i. \quad (4)$$

В нашем примере x_{ij} – количество денежных средств, переводимых из i-ого коммерческого банка к j-му держателю банковской карты (если по условию задачи введен фиктивный коммерческий банк или держатель банковской карты, то необходимо ввести фиктивные переменные, которые обозначаются x_{ij}^{ϕ}).

$F(X)$ – денежные расходы на перевод всех денежных средств (руб.).

Задача сводится к тому, чтобы отыскать неотрицательные значения, при которых целевая функция стремится к минимуму

$$F(x) = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \to \min. \quad (5)$$

Введение фиктивного держателя банковской карты или банка повлечет необходимость формального задания фиктивных тарифов c_{ij}^{ϕ} (реально не существующих) для фиктивного перевода денежных средств. Поскольку нас интересует определение наиболее выгодных реальных переводов денежных средств, то необходимо предусмотреть, чтобы при решении задачи (при нахождении опорных планов) фиктивные переводы не рассматривались до тех пор, пока не будут определены все реальные переводы. Для этого надо фиктивные переводы сделать невыгодными, то есть дорогими, чтобы при поиске решения задачи их рассматривали в самую последнюю очередь. Таким образом, величина фиктивных тарифов должна превышать максимальный из реальных тарифов, используемых в модели, то есть:

$$c_{ij}^{\phi} > \max c_{ij} \left(i = \overline{1,m};\ j = \overline{1,n}\right). \quad (6)$$

При этом необходимо выполнение следующих основных условий:

1) от каждого кредитного банка можно перевести столько денежных средств, сколько у него имеется, то есть сумма искомых переводов от каждого коммерческого банка равна наличию у него денежных средств (условия перевода всех денежных средств из коммерческих банков):

$$\sum_{j=1}^{n} x_{ij} = b_i; \quad (7)$$

2) каждому держателю банковских карт нужно перевести необходимое ему количество денежных средств, то есть сумма искомых переводов равна потребностям держателей банковских карт (условия полного удовлетворения держателей банковских карт):

$$\sum_{i=1}^{m} x_{ij} = a_j; \quad (8)$$

3) условия неотрицательности переменных, исключающие обратные переводы:

$$\forall x_{ij} \geq 0 \quad (i = \overline{1,m};\ \ j = \overline{1,n}). \quad (9)$$

Ограничения модели (7, 8, 9) могут быть выполнены только при сбалансированной задаче.

Кроме основных условий, в транспортных задачах может встретиться ряд дополнительных, ограничивающих количественные связи между отдельными потребителями

и поставщиками. Характер этих ограничений и способы решения задачи при наличии дополнительных ограничений заключаются в следующем:

1) полное отсутствие связи между банком и держателем банковской карты, то есть $x_{ij} = 0$. Это означает, что в данной клетке матрицы искомый объем денежных средств должен быть равен нулю. Оценка переменной завышается на большую величину, обычно обозначаемую буквой М, и «попадание» денежных средств в эту клетку нежелательно, так как целевая функция всегда стремится к минимуму;

2) наличие частной заранее фиксированной связи между банком и держателями банковских карт, то есть $x_{ij} = q$, искомый объем перевода денежных средств от i-го банка к j-му держателю банковских карт должен быть строго равен q. До начала решения задачи от соответствующего банка и держателя банковской карты вычитается величина q, а затем в соответствующую клетку пересечения банка и держателя банковской карты записывается завышенная оценка М и задача решается обычным методом;

3) $x_{ij} > q$, то есть искомый объем денежных средств от i-го банка к j-му держателю банковской карты должен быть не меньше величины q. До начала решения от соответствующего банка и держателя банковской карты вычитается величина q, затем задача решается обычным путем.

Модель транспортной задачи позволяет решать любые задачи, в которых параметры имеют одинаковые единицы измерения. Такие модели называют однопродуктовыми. Целевая функция в модели может стремиться как к min, так и к max, исходя из условия задачи.

3. 2. Решение задач с использованием табличного редактора MS Excel. Экономическая интерпретация полученных результатов.

Три филиала одного коммерческого банка «Форштадт» решили выступить в качестве кредиторов трех предприятий – ОАО «Оренсот», ООО «Триумф» и ЗАО «Стимул». Наличие денежных средств в банках, потребности предприятий в денежных средствах и процентные ставки по кредитам приведены в таблице 2.

Таблица 2 – Исходные данные к задаче

Филиалы КБ «Форштадт»	Предприятия			Наличие денежных средств, тыс.руб.
	ОАО «Оренсот»	ООО «Триумф»	ЗАО «Стимул»	
№1	13	15	21	1000
№2	8	18	7	900
№3	23	19	30	650
Потребность в денежных средствах, тыс. руб.	800	900	850	2550 / 2550

Необходимо определить план выдачи кредитов, удовлетворяющий спрос предприятий и позволяющий банку получить максимальный объем прибыли.

Решение

Проверка сбалансированности задачи

Просуммируем наличие денежных средств у филиалов коммерческого банка «Форштадт», которые необходимо перечислить:

$$\sum_{i=1}^{3} b_i = 1000 + 900 + 650 = 2550.$$

Просуммируем потребность предприятий в денежных средствах:

$$\sum_{j=1}^{3} a_j = 800 + 900 + 850 = 2550.$$

Так как $\sum_{i=1}^{3} b_i = \sum_{j=1}^{3} a_j$, то задача сбалансированная (закрытого типа).

Определение переменных

Обозначим через x_{ij} количество денежных средств, которые будут выданы филиалами коммерческого банка «Форштадт» (i-ым поставщиком) предприятиям (j-му потребителю).

Модель задачи

I. $F(x) = 13x_{11} + 15x_{12} + 21x_{13} + 8x_{21} + 18x_{22} + 7x_{23} + 23x_{31} + 19x_{32} + 30x_{33} \rightarrow \max$

II. $x_{11} + x_{12} + x_{13} = 1000$

$x_{21} + x_{22} + x_{23} = 900$

$x_{31} + x_{32} + x_{33} = 650$

$x_{11} + x_{21} + x_{31} = 800$

$x_{12} + x_{22} + x_{32} = 900$

$x_{13} + x_{23} + x_{33} = 850$

III. $x_{ij} \geq 0$ ($i = \overline{1,3}$; $j = \overline{1,3}$)

Для решения данной задачи в Microsoft Excel необходимо:

1) под запись целевой функции отвести ячейку A1;

2) под запись ограничений – ячейки столбца B (количество ячеек совпадает с количеством ограничений): B1,B2, B3, B4, B5, B6;

3) под запись искомых переменных отвести ячейки столбцов C, D, E (количество предприятий совпадают с количеством столбцов, а количество филиалов коммерческого банка – с количеством строк).

Примечание: искомые переменные x_{ij} будут находится в следующих ячейках:

$x_{11} \rightarrow$ C1 $x_{12} \rightarrow$ D1 $x_{13} \rightarrow$ E1

$x_{21} \rightarrow$ C2 $x_{22} \rightarrow$ D2 $x_{23} \rightarrow$ E2

$x_{31} \rightarrow$ C3 $x_{32} \rightarrow$ D3 $x_{33} \rightarrow$ E3

Порядок выполнения работы

1. Ввести в ячейку A1 формулу целевой функции (рисунок 1):

=13*C1 + 15*D1 + 21*E1+

+ 8*C2 + 18*D2 + 7*E2+

+ 23*C3 + 19*D3 + 30*E3;

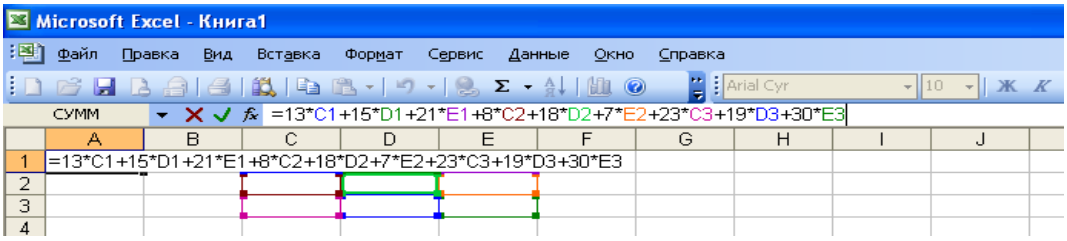


Рисунок 1 – Ввод целевой функции в Excel

2. а) ввести в ячейку B1 левую часть первого ограничения: = C1 + D1 + E1 (рисунок 2);

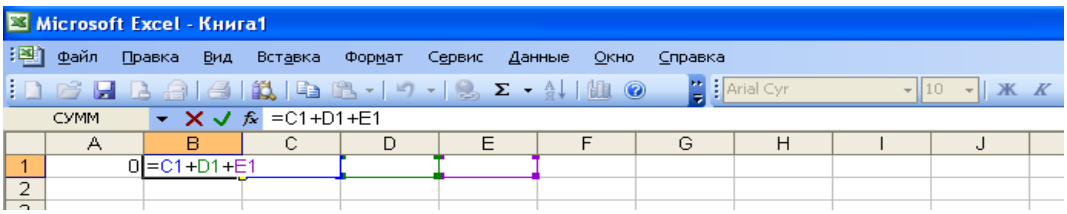


Рисунок 2 – Ввод ограничений в Excel

б) ввести в ячейку B2 левую часть второго ограничения: = C2 + D2 + E2;
в) ввести в ячейку B3 левую часть третьего ограничения: = C3 + D3 + E3;
г) ввести в ячейку B4 левую часть четвертого ограничения: = C1 + C2 + C3;
д) ввести в ячейку B5 левую часть пятого ограничения: = D1+D2+D3;
е) ввести в ячейку B6 левую часть шестого ограничения: = E1+E2+E3 (рисунок 3).

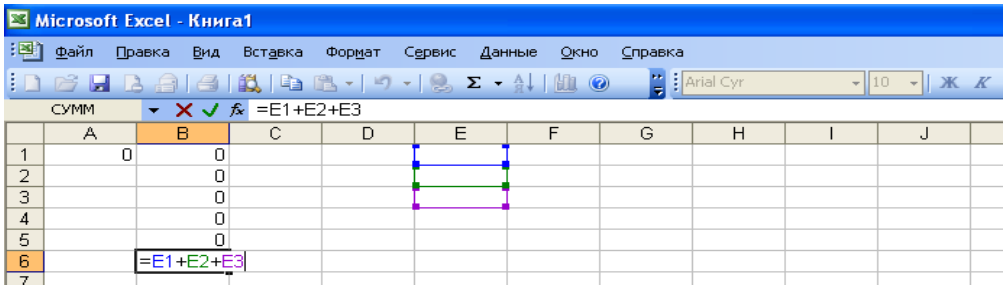


Рисунок 3 – Ввод ограничений в Excel

3. На панели инструментов выбрать опцию «Сервис», а в ней вкладку «Поиск решения».

4. В окне диалога «Поиск решения» в поле ввода «Установить целевую ячейку» нужно ввести ссылку на ячейку A1. Необходимо выбрать способ адресации ячеек в абсолютной системе координат (т.е. указать не A1, а $A$1). Также нужно поступать с другими переменными.

5. В окне диалога «Поиск решения» нужно установить переключатель (рисунок 4).

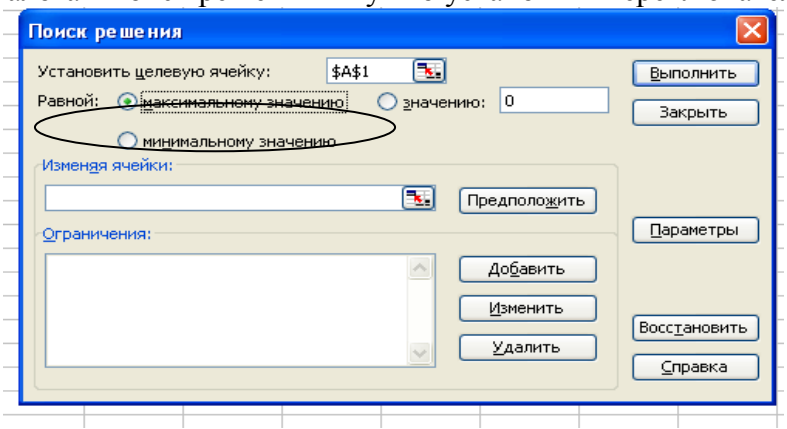


Рисунок 4 – Работа в диалогом окне «Поиск решения»

6. В поле ввода «Изменяя ячейки» нужно указать ссылки на ячейки, содержащие искомые переменные, т.е. диапазон ячеек $C$1:$E$3 (рисунок 5).

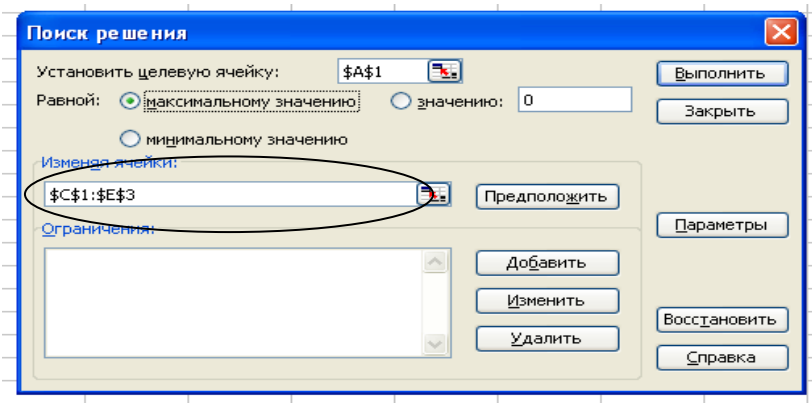


Рисунок 5 – Поле ввода ячеек, обозначающих искомые переменные

7. В поле ввода «Ограничения» при нажатии кнопки «Добавить» появляется окно диалога «Добавить ограничения». В поле ввода «Ссылка на ячейку» вводится $B$1. В поле ввода «Ограничение» вводится = и число 1000. При помощи кнопки «Добавить» таким же образом вводятся все остальные ограничения (ячейки $B$2:$B$6) (рисунок 6).

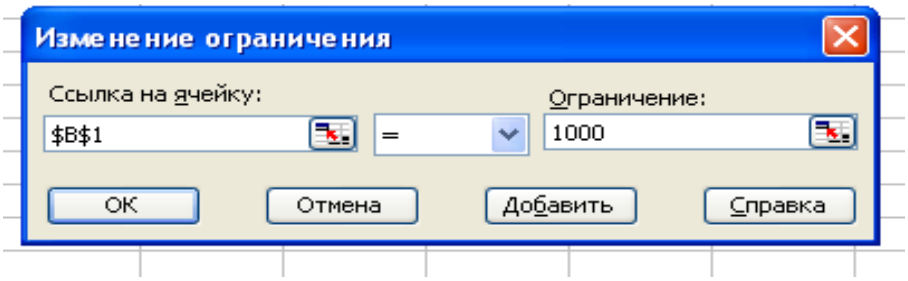
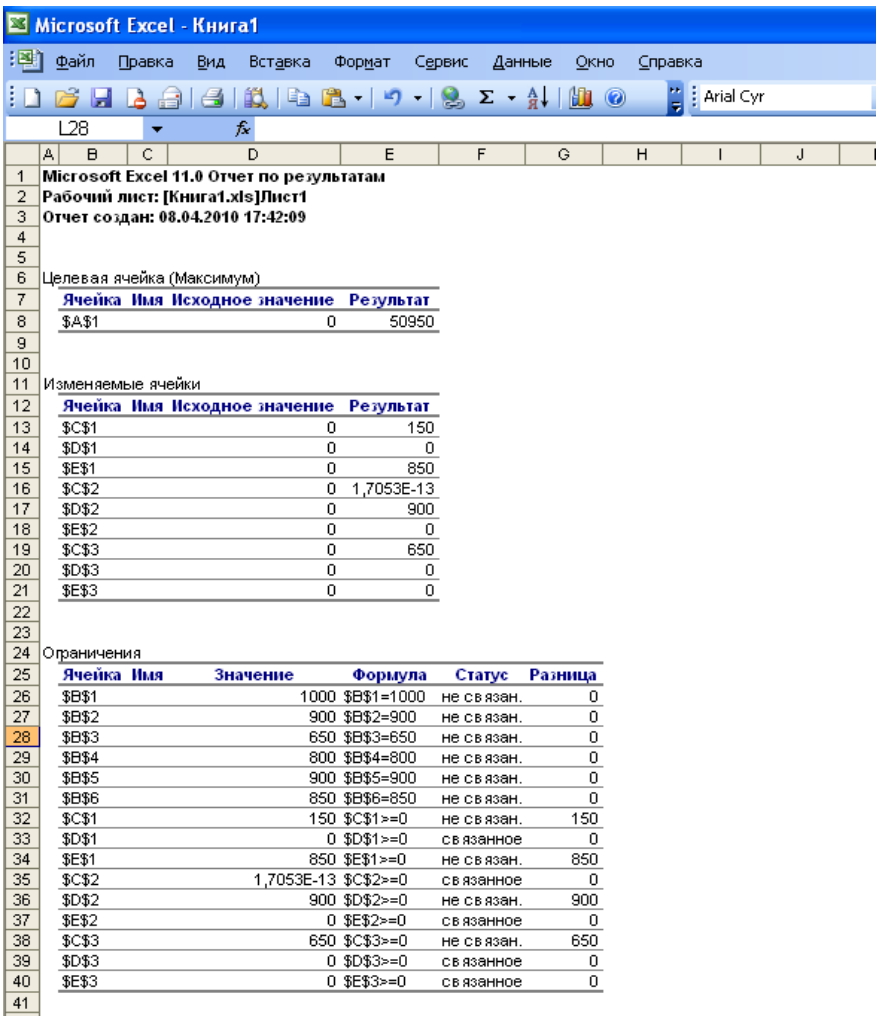


Рисунок 6 – Диалоговое окно «Добавление ограничения»

8. Для ввода ограничений на неотрицательность искомых переменных в окне диалога «Добавить ограничения» в поле ввода «Ссылка на ячейку» нужно ввести ссылку на ячейку $C$1, а в поле ввода «Ограничения» нужно ввести >= и число 0. При помощи кнопки «Добавить» таким же образом вводятся условия неотрицательности оставшихся искомых переменных. Либо выделяется диапазон ячеек $C$1:$E$3 и задается >= и число 0. после ввода последнего ограничения нажмите «ОК».

9. Затем добавим условие целочисленности. Для этого в поле ввода «Ограничения» выделяем диапазон ячеек $C$1:$E$3 и задаем «цел. числа», затем нажимаем «ОК».

10. После нажатия кнопки «Выполнить» Excel рассчитывает результат и открывает окно диалога «Результаты поиска решения». В этом диалоге в окне «Тип отчета» нужно выбрать «Результаты» и нажать «ОК». Перед листом, где записана постановка задачи, будет вставлен лист «Отчет по результатам 1», а на экране будет выдан результат решения задачи (рисунки 7 и 8).



Microsoft Excel - Книга1

Файл Правка Вид Вставка Формат Сервис Данные Окно Справка

Arial Cyr

L28

Microsoft Excel 11.0 Отчет по результатам
Рабочий лист: [Книга1.xls]Лист1
Отчет создан: 08.04.2010 17:42:09

Целевая ячейка (Максимум)

Ячейка	Имя	Исходное значение	Результат
$A$1		0	50950

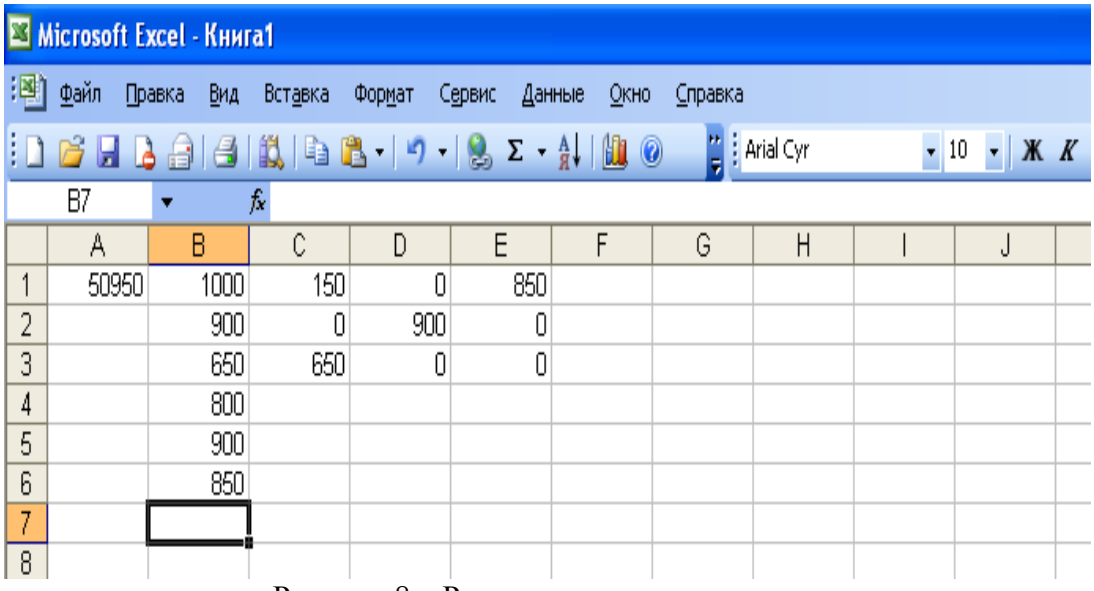
Изменяемые ячейки

Ячейка	Имя	Исходное значение	Результат
$C$1		0	150
$D$1		0	0
$E$1		0	850
$C$2		0	1,7053E-13
$D$2		0	900
$E$2		0	0
$C$3		0	650
$D$3		0	0
$E$3		0	0

Ограничения

Ячейка	Имя	Значение	Формула	Статус	Разница
$B$1		1000	$B$1=1000	не связан.	0
$B$2		900	$B$2=900	не связан.	0
$B$3		650	$B$3=650	не связан.	0
$B$4		800	$B$4=800	не связан.	0
$B$5		900	$B$5=900	не связан.	0
$B$6		850	$B$6=850	не связан.	0
$C$1		150	$C$1>=0	не связан.	150
$D$1		0	$D$1>=0	связанное	0
$E$1		850	$E$1>=0	не связан.	850
$C$2		1,7053E-13	$C$2>=0	связанное	0
$D$2		900	$D$2>=0	не связан.	900
$E$2		0	$E$2>=0	связанное	0
$C$3		650	$C$3>=0	не связан.	650
$D$3		0	$D$3>=0	связанное	0
$E$3		0	$E$3>=0	связанное	0

Рисунок 7 – Отчет по результатам



Microsoft Excel - Книга1

Файл Правка Вид Вставка Формат Сервис Данные Окно Справка

Arial Cyr 10

B7

	A	B	C	D	E	F	G	H	I	J
1	50950	1000	150	0	850					
2		900	0	900	0					
3		650	650	0	0					
4		800								
5		900								
6		850								
7										
8										

Рисунок 8 – Результаты решения задачи

Ответ:

$$X = \begin{pmatrix} 150 & 0 & 850 \\ 0 & 900 & 0 \\ 650 & 0 & 0 \end{pmatrix};$$

$F(x) = 50950$.

Получение максимального объема прибыли филиалами коммерческого банка «Форштадт» может быть достигнуто путем наиболее оптимальной выдачи кредитов потребителям ОАО «Оренсот», ООО «Триумф» и ЗАО «Стимул». Целесообразный вариант выдачи денежных средств предусматривает следующие комбинации:

– филиал №1 выдаст денежные средства предприятию ОАО «Оренсот» в размере 150 тыс.руб. и предприятию ЗАО «Стимул» в размере 850 тыс.руб.;

– филиал №2 предоставит кредит только предприятию ООО «Триумф» в размере 900 тыс.руб.;

– филиал №3 выдаст денежные средства только предприятию ОАО «Оренсот» в размере 650 тыс.руб. Данный план выданных кредитов позволит удовлетворить спрос предприятий и получить банку максимальный объем прибыли в размере 50 950 тыс. руб.

Задания

Задача 1

Три банка, отделения, подотчетные одному головному офису, решили выступить в качестве кредиторов трех сельскохозяйственных предприятий. Известны запасы денежных средств, потребность предприятий в кредитных ресурсах и ставки по процентам. Данные представлены в таблице 3.

Таблица 3 – Исходные данные к задаче 1

Банки	Сельскохозяйственные предприятия			Наличие денежных средств, тыс.руб.
	1	2	3	
1	15	17	23	970
2	19	14	20	830
3	24	21	18	1200
Потребность в кредитных ресурсах, тыс.руб.	960	1210	830	

Необходимо определить план выданных кредитов, обеспечивающий максимально выгодные условия (т.е. оптимальные кредитные суммы при минимальных процентных ставках) для кредиторов.

Задача 2

Трем акционерам АО «Урал» необходимо выкупить некоторое количество акций трех предприятий. Общая стоимость акций, потребности в них каждого акционера и затраты на их приобретение приведены в таблице 4.

Таблица 4 – Исходные данные к задаче 2

Предприятие	Акционеры ЗАО «Урал»			Наличие акций, тыс.руб.
	Иванов П.И.	Петров В.Н.	Сидоров А.С.	
«Закат»	200	500	50	800
«Рассвет»	300	100	400	700
«Заря»	150	250	350	900
Потребности в акциях, тыс.руб.	850	600	950	

Определите, на какую сумму следует купить каждому акционеру акций каждого предприятия, чтобы общая сумма покупки была минимальной.

Задача 3

Требуется получить кредит в трех банках соответственно трем заемщикам одной организации. Размер кредита, выданного каждым банком и потребность в них каждого заемщика, приведены в таблице 5.

Таблица 5 – Исходные данные к задаче 3

Банки	Заемщики			Наличие кредита, тыс.руб.
	1	2	3	
1	20	22	28	950
2	14	24	17	850
3	29	26	37	600
Потребность в кредите, тыс.руб.	750	850	800	

Определите, какому банку следует удовлетворять спрос заемщика, чтобы общая сумма объема кредита была минимальной.

Задача 4

В Ассоциацию инвесторов России входят три крупнейшие инвестиционные компании: «Рус-Инвест», «ИнКом», «Инвест-Гарант». Ассоциация разработала программу инвестирования, предусматривающую вложение инвестиций в четыре компании. Доходность вложений, потребность в инвестициях и их наличие представлены в таблице 6.

Таблица 6 – Исходные данные к задаче 4

	ООО «Русь»	ОАО «Галина»	ЗАО «Луч»	ЗАО «Ява»	Наличие инвестиций, млн.руб.
«Рус-Инвест»	9	7	12	5	50
«ИнКом»	12	15	8	10	60
«Инвест-Гарант»	5	10	12	15	78
Потребность в инвестициях, млн.руб.	62	31	42	45	

Определить оптимальное распределение инвестиций для получения максимального дохода.

Задача 5

Три предприятия хотят получить кредит в трех филиалах одного банка на развитие производства. Наличие денежных средств в банке, потребности предприятий в денежных средствах и срок кредитования приведены в таблице 7.

Таблица 7 – Исходные данные к задаче 5

Филиалы банков	Предприятия			Наличие денежных средств, тыс.руб.
	1	2	3	
1	13	15	21	1000
2	8	18	7	900
3	23	19	30	650
Потребность денежных средств, тыс.руб.	800	900	850	

Нужно определить, какой филиал банка максимально удовлетворит спрос потребителя, чтобы общая сумма объема выданных средств была минимальной и целесообразна.

Задача 6

Три предпринимателя приобретают акции трех компаний одной отрасли. Количество акций компании, потребность предпринимателей в акциях каждой компании, а так же стоимость одной акции представлены в таблице 8.

Таблица 8 – Исходные данные к задаче 6

Акции компании	Предприниматели			Количество акций, шт.
	1	2	3	
1	15	17	23	900
2	9	19	8	800
3	24	21	32	550
Потребность в акциях, шт.	700	800	850	

Необходимо определить, акции какой из компаний следует покупать предпринимателям, чтобы суммарные затраты на приобретение были минимальны.

Задача 7

Необходимо обменять иностранную валюту на рубли. Количество валюты, которое может выдавать обменный пункт, потребности и курсы валют представлены в таблице 9.

Таблица 9 – Исходные данные к задаче 7

Обменный пункт	Валюта			Количество, тыс.руб.
	Доллар	Евро	Гривна	
А	32,5	44,6	21	100
Б	32,6	44,7	7	150
В	32,55	44,6	30	50
Потребности, тыс.руб.	170	60	70	

Определить, какое количество и в каком обменном пункте следует обменять валюты, чтобы общая сумма полученных пунктом рублей была минимальной.

3. Контрольная работа.

2.3 Лабораторная работа №8, 9, 10, 11 (8 часов).

Тема: «Межотраслевые балансовые модели. Решение задач в MS Excel»

2.3.1 Цель работы: Проанализировать принципиальную схему межотраслевого баланса и способы решения данных задач в MS Excel

2.3.2 Задачи работы:

1. Анализ принципиальной схемы межотраслевого баланса

2. Изучить способы решения данных задач в MS Excel

2.3.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. OpenOffice

2. Microsoft Office Excel

2.3.4 Описание (ход) работы:

1. Принципиальная схема межотраслевого баланса. Межотраслевые балансовые модели в анализе экономических показателей.

Межотраслевой баланс представляет собой таблицу, описывающую баланс производства и распределения продукции в народном хозяйстве. Межотраслевой баланс (МОБ) производства и распределения продукции строится по схеме «затраты-выпуск».

Пусть рассматривается экономическая система, состоящая из n взаимосвязанных отраслей производства. Продукция каждой отрасли частично идет на внешнее потребление (конечный продукт), а частично используется в качестве сырья, полуфабрикатов или других средств производства в других отраслях, в том числе и в данной. Эту часть продукции называют *производственным потреблением.* Поэтому каждая из рассматриваемых отраслей выступает и как производитель продукции (первый столбец таблицы 3.1) и как ее потребитель (первая строка таблицы 1).

Обозначим через X_i валовой выпуск продукции i-й отрасли за планируемый период и через Y_i – конечный продукт, идущий на внешнее для рассматриваемой системы потребление (средства производства других экономических систем, потребление населения, образование запасов и т.д.).

Таким образом, разность $X_i - Y_i$ составляет часть продукции i-й отрасли, предназначенную для внутрипроизводственного потребления. Будем в дальнейшем полагать, что баланс составляется не в натуральном, а в стоимостном разрезе.

Обозначим через x_{ik} часть продукции i-й отрасли, которая потребляется k-й отраслью, для обеспечения выпуска ее продукции в размере x_k.

В МОБ выделяют четыре части, имеющие различное экономическое содержание, они называются квадрантами баланса.

Первый квадрант МОБ – это шахматная таблица межотраслевых материальных связей. Показатели, помещенные на пересечениях строк и столбцов, представляют собой величины межотраслевых потоков продукции и в общем виде обозначаются x_{ij}, где i и j – соответственно номера отраслей производящих и потребляющих. Таким образом, первый квадрант по форме представляет собой квадратную матрицу порядка n, сумма всех элементов которой равняется годовому фонду возмещения затрат средств производства в материальной сфере.

Во втором квадранте представлена конечная продукция всех отраслей материального производства. При этом под конечной продукцией понимается продукция, выходящая из сферы производства в область конечного использования (на потребление и накопление). Сюда включается накопление и возмещение выбытия основных фондов, прирост запасов, личное потребление населения, расходы продукции на содержание государственного

аппарата и оборону, затраты на обслуживание населения (просвещение, здравоохранение и т.д.). В таблице 1 этот раздел дан укрупнено в виде одного столбца величин Y_i.

Таблица 1 – Принципиальная схема межотраслевого баланса (МОБ)

Производящие отрасли	Потребляющие отрасли					Конечный продукт	Валовой продукт
	1	2	3	…	n		
1	x_{11}	x_{12}	x_{13}	…	x_{1n}	Y_1	X_1
2	x_{21}	x_{22}	x_{23}	…	x_{2n}	Y_2	X_2
3	x_{31}	x_{32}	x_{33}	…	x_{3n}	Y_3	X_3
…	…	…	…	…	…		
n	x_{n1}	x_{n2}	x_{n3}		x_{nn}	Y_n	X_n
Амортизация	c_1	c_2	c_3	…	c_n	$\sum_{i=1}^{n} X_i = \sum_{j=1}^{n} X_j$	
Оплата труда	v_1	v_2	v_3	…	v_n		
Чистый доход	m_1	m_2	m_3	…	m_n		
Валовой продукт	X_1	X_2	X_3	…	X_n		

Третий квадрант МОБ также характеризует национальный доход, но со стороны его стоимостного состава как сумму чистой продукции и амортизации. Чистая продукция понимается при этом как сумма оплаты труда и чистого дохода отраслей. Сумма амортизации (c_j) и чистой продукции ($v_j + m_j$) некоторой j-ой отрасли будем называть условно чистой продукцией этой отрасли и обозначать в дальнейшем Z_j.

Четвертый квадрант баланса находится на пересечении столбцов второго квадранта (конечной продукции) и строк третьего квадранта (условно чистой продукции). Этим определяется содержание квадранта: он отражает конечное распределение и использование национального дохода. В результате перераспределения первоначально созданного национального дохода образуются конечные доходы населения, предприятий, государства. Данные четвертого квадранта важны для отражения в межотраслевой модели баланса доходов и расходов населения, источников финансирования капиталовложений, текущих затрат непроизводственной сферы, для анализа общей структуры конечных доходов по группам потребителей. Общий итог четвертого квадранта, также как второго и третьего, должен быть равен созданному за год национальному доходу.

Таким образом, в целом МОБ в рамках единой модели объединяет балансы отраслей материального производства, баланс совокупного общественного продукта, балансы национального дохода, финансовый, баланс доходов и расходов населения.

Итог материальных затрат любой потребляющей отрасли и ее условно чистой продукции равен валовой продукции этой отрасли. Данный вывод можно записать в виде соотношения:

$$X_j = \sum_{i=1}^{n} x_{ij} + Z_j, j = 1,\ldots, n. \quad (1)$$

Напомним, что величина условно чистой продукции Z_j равна сумме амортизации, оплаты труда и чистого дохода j-й отрасли.

Валовая продукция той или иной отрасли равна сумме материальных затрат потребляющих ее продукцию отраслей и конечной продукции данной отрасли:

$$X_i = \sum_{j=1}^{n} x_{ij} + Y_i, i = 1,\ldots, n. \quad (2)$$

Формула описывает систему из n уравнений, которые называются уравнениями распределения продукции отраслей материального производства по направлениям использования.

Просуммируем по всем отраслям уравнения 1, в результате получим:

$$\sum_{j=1}^{n} X_j = \sum_{j=1}^{n}\sum_{i=1}^{n} x_{ij} + \sum_{j=1}^{n} Z_j.$$

Аналогичное суммирование уравнений 3.2 дает:

$$\sum_{i=1}^{n} X_i = \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij} + \sum_{i=1}^{n} Y_i. \quad (3)$$

Левые части обоих равенств раны, так как представляют собой весь валовой общественный продукт. Первые слагаемые правых частей этих равенств также равны, их величина равна итогу первого квадранта. Следовательно, должно соблюдаться соотношение:

$$\sum_{j=1}^{n} Z_j = \sum_{i=1}^{n} Y_i. \quad (4)$$

Левая часть данного уравнения есть сумма третьего квадранта, а первая часть – итог второго квадранта.

Дадим понятие и представим расчет коэффициентов прямых и полных материальных затрат.

Величины a_{ij} называются *коэффициентами прямых материальных затрат* и рассчитываются следующим образом:

$$a_{ij} = \frac{x_{ij}}{X_j},\ i = 1,\ldots, n. \quad (5)$$

Существует два основных определения коэффициентов прямых и полных материальных затрат.

Определение 1. Коэффициенты прямых материальных затрат показывают среднюю величину затрат продукции одной отрасли на производство единицы продукции другой отрасли.

С учетом формулы 3.5 систему уравнений баланса можно переписать в виде:

$$X_i = \sum_{j=1}^{n} a_{ij} X_j + Y_i,\ i = 1,\ldots, n. \quad (6)$$

Система уравнений 6 в матричной форме примет вид:

$$X = AX + Y. \quad (7)$$

Система уравнений 6, или в матричной форме 7, называется экономико-математической моделью МОБ (моделью Леонтьева, моделью «затраты-выпуск»). С помощью этой модели можно выполнить три варианта расчетов:

1) задав в модели величины валовой продукции каждой отрасли (X_i), можно определить объемы конечной продукции каждой отрасли (Y_i):

$$Y = (E - A)\,X; \quad (8)$$

2) задав величины конечной продукции всех отраслей (Y_i), можно определить величины валовой продукции каждой отрасли (X_i):

$$X = (E - A)^{-1}Y; \quad (9)$$

3) для ряда отраслей задав величины валовой продукции, а для всех остальных отраслей задав объемы конечной продукции, можно найти величины конечной продукции первых отраслей и объемы валовой продукции вторых. В этом варианте расчета удобнее пользоваться не матричной формой модели 7, а системой линейных уравнений 6. В формулах 8 и 9 E обозначает единичную матрицу n-го порядка, а $(E - A)^{-1}$ обозначает матрицу, обратную матрице $(E - A)$. Если определитель матрицы $(E - A)$ не равен нулю, т.е. эта матрица невырожденная, то обратная к ней матрица существует. Обозначим эту обратную матрицу через B, тогда систему уравнений в матричной форме 9 можно записать в виде:

$$X = BY. \quad (9')$$

Элементы матрицы B будем обозначать через b_{ij}, тогда из матричного уравнения 9' для любой i-й отрасли можно получить следующее соотношение:

$$X_i = \sum_{j=1}^{n} b_{ij} Y_i, i = \overline{1, n}. \quad (10)$$

Из соотношений 10 следует, что валовая продукция выступает как взвешенная сумма величин конечной продукции, причем весами являются коэффициенты b_{ij}, которые показывают, сколько всего нужно произвести продукции i-й отрасли для выпуска в сферу конечного использования единицы продукции j-й отрасли. В отличии от коэффициентов прямых затрат a_{ij} коэффициенты b_{ij} называются *коэффициентами полных материальных затрат*
и включают в себя как прямые, так и косвенные затраты всех порядков. Если прямые затраты отражают количество средств производства, израсходованных непосредственно при изготовлении данного продукта, то косвенные относятся к предшествующим стадиям производства и входят в производство продукта не прямо, а через другие (промежуточные) средства производства.

Определение 2. Коэффициенты полных материальных затрат b_{ij} характеризуют полные затраты продукции одной отрасли на единицу продукции другой, или показывает какое количество продукции одной отрасли нужно произвести, чтобы получить единицу конечной продукции другой отрасли.

Коэффициенты прямых затрат по определению являются неотрицательными, следовательно, матрица A в целом может быть названа неотрицательной: $A \geq 0$. Так как процесс воспроизводства нельзя было бы осуществлять, если бы для собственного воспроизводства в отрасли затрачивалось большее количество продукта, чем создавалось, то очевидно, что диагональные элементы матрицы A меньше единицы: $a_{ij} < 1$.

Дадим понятие продуктивности матрицы коэффициентов прямых материальных затрат.

Будем называть неотрицательную матрицу A продуктивной, если существует такой неотрицательный вектор $X \geq 0$, что

$$X > AX. \quad (11)$$

Очевидно, что условие 11 означает существование положительного вектора конечной продукции $Y > 0$ для модели МОБ 3.7.

Для того чтобы матрица *коэффициентов прямых затрат* A была *продуктивной*, необходимо и достаточно, чтобы выполнялось одно из перечисленных ниже условий:

1) *матрица* $(E - A)$ *неотрицательно обратима,* т.е. существует обратная матрица $(E-A)^{-1} \geq 0$;

2) *матричный ряд* $E + A + A^2 + A^3 + ... = \sum_{k=0}^{\infty} A^k$ *сходится,* причем его сумма равна обратной матрице $(E-A)^{-1} \geq 0$;

3) *наибольшее по модулю собственное значение* λ матрицы A, т.е. решение характеристического уравнения $|\lambda E - A| = 0$, *строго меньше единицы;*

4) *все главные миноры* матрицы $(E - A)$, т.е. определители матриц, образованные элементами первых строк и первых столбцов этой матрицы, порядка от 1 до n, *положительны.*

Более простым, но только достаточным признаком продуктивности матрицы A является ограничение на величину ее нормы, т.е. на величину наибольшей суммы элементов матрицы A в каждом столбце. Если норма матрицы A строго меньше единицы, то эта матрица продуктивна. Заметим, что данное условие является только достаточным, и матрица A может оказаться продуктивной и в случае, когда ее норма больше единицы.

2. Решение задач с использованием табличного редактора MS Excel. Экономическая интерпретация полученных результатов.

Задания

Задача 1

Закончите составление схемы отчетного баланса по имеющимся данным.

Таблица 2 – Исходные данные к задаче 1

Потребляющие отрасли	Производящие отрасли		Конечный продукт Y_i	Валовой продукт X_i
	1	2		
1	20		30	
2	60	10		
Условно чистая продукция Z_j		70		
Валовой продукт X_j	100			

Решение

1) если $i = j$, то $X_j = X_i$ => если $i = j = 1$, то $X_j = X_i = 100$.

2) $X_i = \sum_{j=1}^{n} x_{ij} + Y_i$ => $x_{12} = X_1 - Y_1 - x_{11}$, $x_{12} = 100 - 30 - 20 = 50$.

3) $X_j = \sum_{i=1}^{n} x_{ij} + Z_j$ => $X_2 = \sum_{i=1}^{n} x_{i2} + Z_2$, $X_2 = 60 + 10 + 70 = 130$,

=> $Z_1 = X_1 - \sum_{i=1}^{2} x_{i1}$, $Z_1 = 100 - (20 + 60) = 20$.

4) $X_i = \sum_{j=1}^{n} x_{ij} + Y_i$ => $Y_2 = X_2 - \sum_{i=1}^{2} x_{2j}$, $Y_2 = 130 - (60 + 10) = 60$.

Ответ

Таблица 3 – Результативная матрица задачи 1

Потребляющие отрасли	Производящие отрасли		Конечный продукт Y_i	Валовой продукт X_i
	1	2		
1	20	**50**	30	**100**
2	60	10	**60**	**130**
Условно чистая продукция Z_j	**20**	70		
Валовой продукт X_j	100	**130**		

Задача 2

Закончите составление схемы отчетного баланса по имеющимся данным в таблице 4.

Таблица 4 – Исходные данные к задаче 2

Потребляющие отрасли	Производящие отрасли		Конечный продукт Y_i	Валовой продукт X_i
	1	2		
1	74	176	120	370
2	222	35	200	457
Условно чистая продукция Z_j	74	245		
Валовой продукт X_j	370	457		

Задача 3

Закончите составление схемы отчетного баланса по имеющимся данным в таблице 3.

Таблица 3 – Исходные данные к задаче 3

Потребляющие отрасли	Производящие отрасли		Конечный продукт Y_i	Валовой продукт X_i
	1	2		
1	30	100	170	300
2	55	165	230	450
Условно чистая продукция Z_j	215	185		
Валовой продукт X_j	300	450		

Задача 4

Закончите составление схемы отчетного баланса по имеющимся данным в таблице 4.

Таблица 4 – Исходные данные к задаче 4

Потребляющие отрасли	Производящие отрасли			Конечный продукт Y_i	Валовой продукт X_i
	1	2	3		
1	15	33	50	190	288
2	28	27	27	233	315
3	15	17	19	121	172
Условно чистая продукция Z_j	230	238	76		
Валовой продукт X_j	288	315	172		

Задача 5

Закончите составление схемы отчетного баланса по имеющимся данным в таблице 5.

Таблица 5 – Исходные данные к задаче 5

Потребляющие отрасли	Производящие отрасли			Конечный продукт Y_i	Валовой продукт X_i
	1	2	3		
1	67	112	124	712	1015
2	103	57	129	811	1100
3	200	78	35	675	988
Условно чистая продукция Z_j	645	853	700		
Валовой продукт X_j	1015	1100	988		

Задача 6

Используя данные баланса (таблица 6), определите объемы производства валовой продукции, коэффициенты прямых и полных материальных затрат.

Таблица 6 – Исходные данные к задаче 6

Производящие отрасли	Потребляющие отрасли		Конечный продукт
	1	2	
1	90	100	50
2	50	110	40

Решение (способ 1)

1) определяем объемы производства валовой продукции (X_i) по формуле:

$$X_i = \sum_{j=1}^{n} x_{ij} + Y_i \text{ , } i = 1, \ldots, n$$

$X_1 = 90 + 100 + 60 = 250$;
$X_2 = 50 + 110 + 40 = 200$.

2) вычислим коэффициенты прямых затрат (a_{ij}) по формуле:

$$a_{ij} = \frac{x_{ij}}{X_j}, \; i = 1, \ldots, n, \; j = 1, \ldots, n.$$

$a_{11} = 90 : 250 = 0{,}36$; $a_{12} = 100 : 200 = 0{,}5$;
$a_{21} = 50 : 250 = 0{,}2$; $a_{22} = 110 : 200 = 0{,}55$.

3) рассчитаем матрицу полных материальных затрат по формуле:
$B = (E - A)^{-1}$.
а) найдем матрицу $E - A$

$$E - A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \begin{pmatrix} 0{,}36 & 0{,}5 \\ 0{,}2 & 0{,}55 \end{pmatrix} = \begin{pmatrix} 0{,}64 & -0{,}5 \\ -0{,}2 & 0{,}45 \end{pmatrix}$$

б) рассчитаем определитель матрицы
Определителем квадратной матрицы 2-го порядка A называется число $a_{11} \cdot a_{22} - a_{12} \cdot a_{21}$. Определитель обозначается $\Delta(A)$ или $\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}$

$\Delta(E - A) = 0{,}64 \cdot 0{,}45 - (-0{,}5) \cdot (-0{,}2) = 0{,}288 - 0{,}1 = 0{,}188$;

в) вместо каждого элемента матрицы поставим его *алгебраическое дополнение*: $\begin{pmatrix} 0{,}45 & 0{,}2 \\ 0{,}5 & 0{,}64 \end{pmatrix}$.

Алгебраическим дополнением каждого элемента определителя называется *минор* этого элемента, умноженный на $(-1)^s$, где s – сумма номеров строки и столбца, на пересечении которых расположен этот элемент.

Минором некоторого элемента определителя называется определитель, получаемый из данного определителя вычеркиванием строки и столбца, на пересечении которых расположен этот элемент.

г) полученную матрицу транспонируем

$$\begin{pmatrix} 0{,}45 & 0{,}5 \\ 0{,}2 & 0{,}64 \end{pmatrix};$$

д) каждый элемент полученной матрицы делим на определитель исходной матрицы и получаем матрицу обратную данной:

$$B = (E - A)^{-1} = \begin{pmatrix} 2{,}39 & 2{,}66 \\ 1{,}06 & 3{,}40 \end{pmatrix}.$$

В качестве проверки можно рассчитать матрицу X:

$$X = BY = \begin{pmatrix} 2{,}39 & 2{,}66 \\ 1{,}06 & 3{,}40 \end{pmatrix} \cdot \begin{pmatrix} 60 \\ 40 \end{pmatrix} = \begin{pmatrix} 250 \\ 200 \end{pmatrix},$$

$X_1 = 2{,}39 \cdot 60 + 2{,}66 \cdot 40 = 249{,}8$;
$X_2 = 1{,}06 \cdot 60 + 3{,}40 \cdot 40 = 199{,}6$.

Решение (способ 2 с использованием MS Excel)

Применение специализированных программ или более доступных, таких как Excel облегчают выполнение представленных расчетов. Поэтому рассмотрим решение этого примера в среде Excel. Для расчетов нами будут использоваться такие функции как МОБР (расчет обратной матрицы) МУММНОЖ (умножение матриц).

Заносим исходные данные в электронную таблицу Excel (рисунок 1).

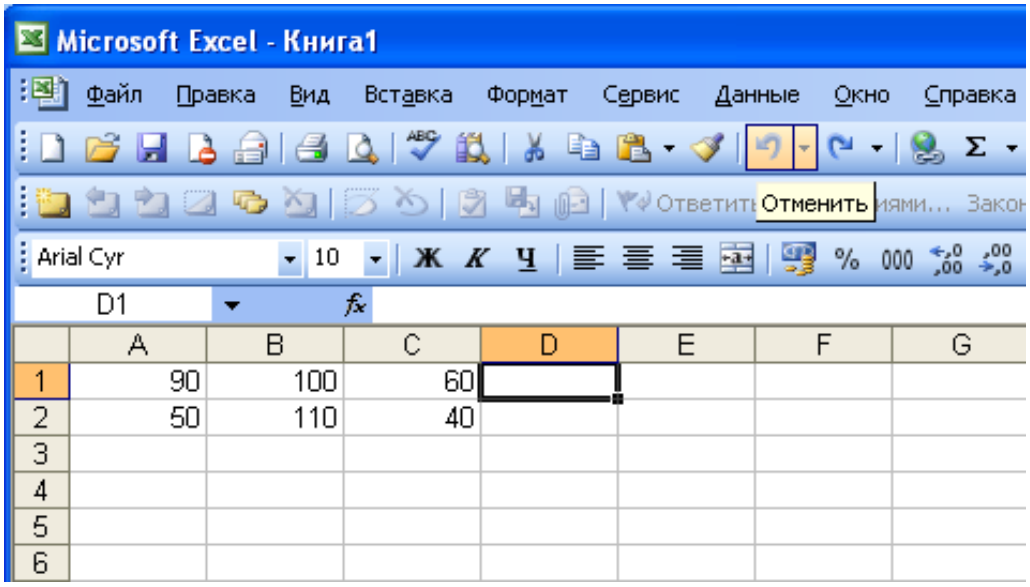


Рисунок 1 – Исходные данные

1) Определяем объемы производства валовой продукции (X_i) по формуле:

$$X_i = \sum_{j=1}^{n} x_{ij} + Y_i \ , \ i = 1,\ldots,n$$

Для этого в ячейку D1 заносим формулу: =СУММА (A1:C1), в ячейку D2: =СУММА (A2:C2) (рисунок 2).

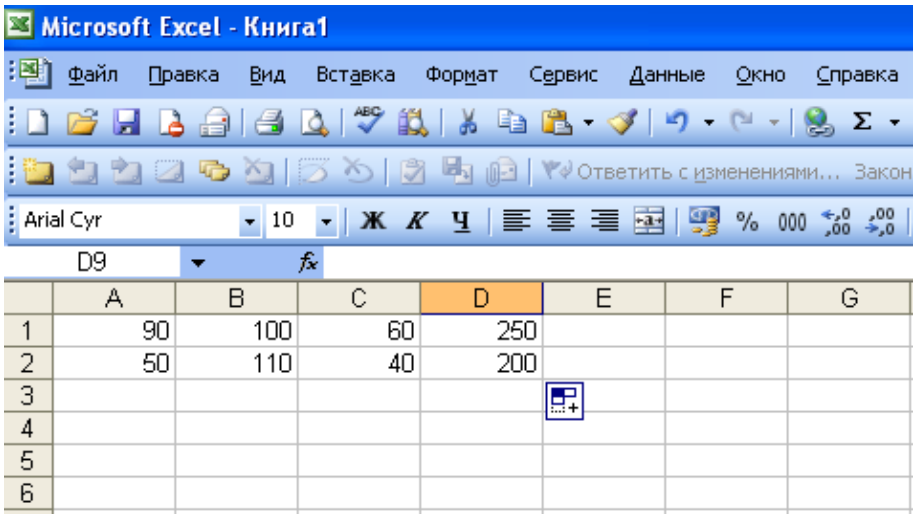


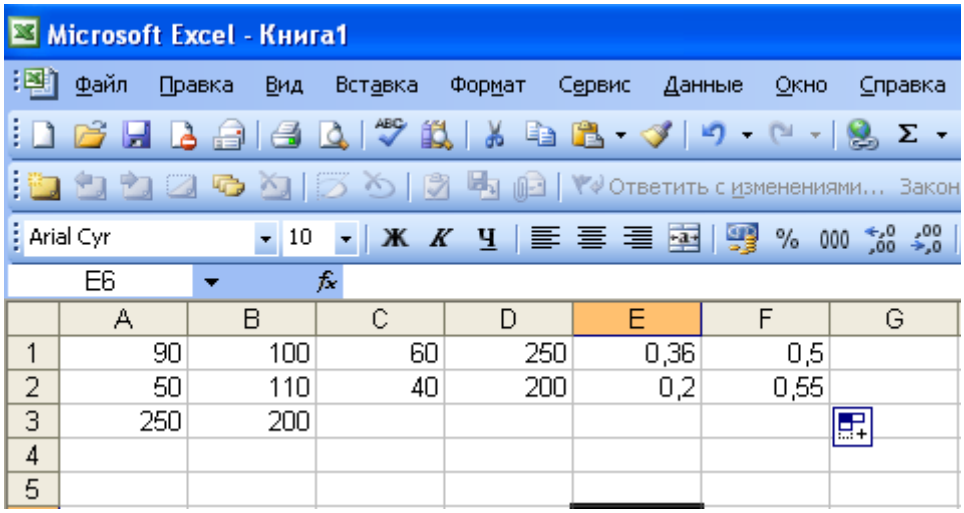
Рисунок 2 – Расчет X_i

2) Вычислим коэффициенты прямых затрат (a_{ij}) по формуле:

$$a_{ij} = \frac{x_{ij}}{X_j} \ , \ i = 1,\ldots, n, \ j = 1,\ldots, n.$$

Для этого в ячейки A3 и B3 переносим значения X_i, рассчитанные в столбце D (можно набрать с клавиатуры, можно использовать функцию «Правка →специальная вставка…→ вставить значения, транспонировать»).

В ячейку E1 записываем формулу: A1/A$3, копируем эту формулу в диапазоне E1:F2. Результатом будет являться матрица прямых коэффициентов A (рисунок 3).



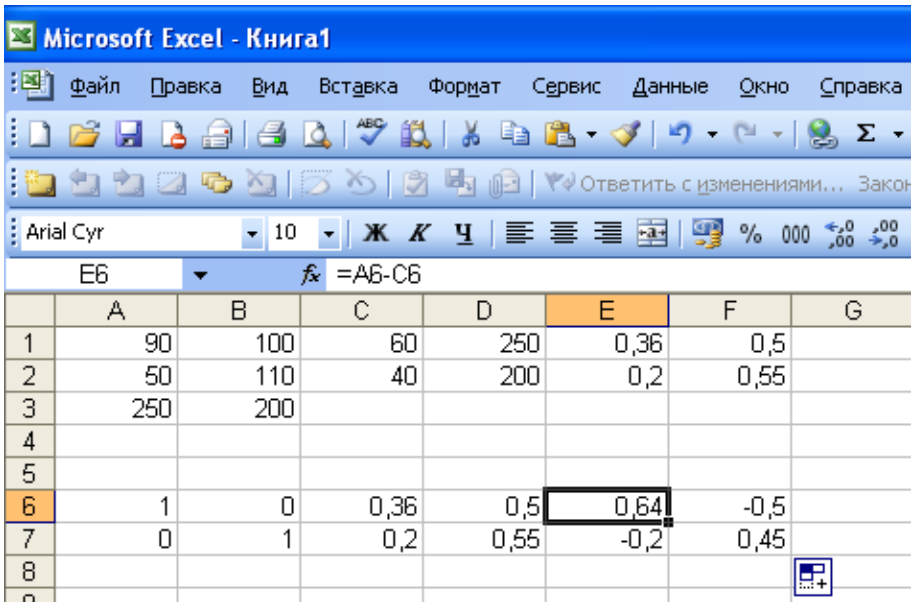
	A	B	C	D	E	F	G
1	90	100	60	250	0,36	0,5	
2	50	110	40	200	0,2	0,55	
3	250	200					
4							
5							

Рисунок 3 – Расчет матрицы коэффициентов прямых затрат

3) Рассчитаем матрицу полных материальных затрат по формуле:

$B = (E - A)^{-1}$

а) найдем матрицу $(E - A)$ (рисунок 4), в диапазоне A6:B7 запишем единичную матрицу и в диапазоне C6:D7 матрицу A. В ячейку E6 запишем формулу: = A1–C1, копируем эту формулу в диапазоне E6:F7, результатом является матрица $(E - A)$.



	A	B	C	D	E	F	G
1	90	100	60	250	0,36	0,5	
2	50	110	40	200	0,2	0,55	
3	250	200					
4							
5							
6	1	0	0,36	0,5	0,64	-0,5	
7	0	1	0,2	0,55	-0,2	0,45	
8							

Рисунок 4 – Расчет матрицы $(E - A)$

б) найдем матрицу обратную $(E - A)$, для этого на листе Excel выделим диапазон G6:H7. Дадим команду «Вставка → Функция...». В открывшемся окне «Мастер функций» необходимо выбрать категорию «Математические», из математических – МОБР (рисунок 5).

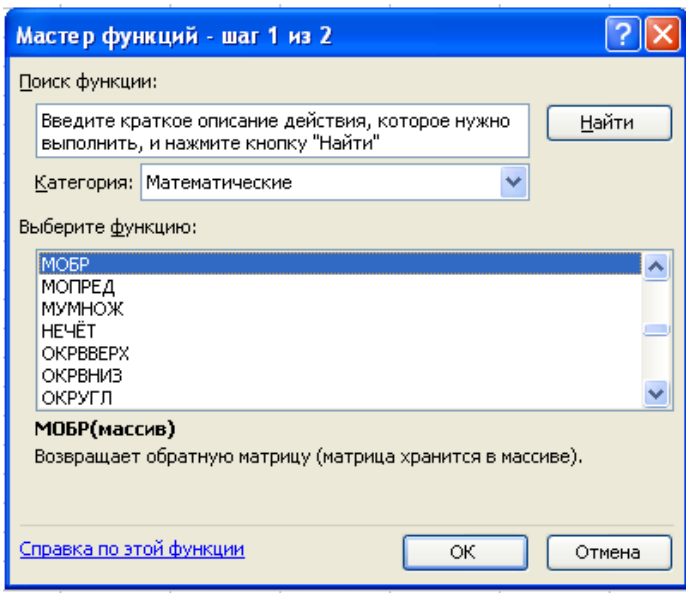


Рисунок 5 – Окно «Мастер функций»

Нажмите ОК. Откроется окно « Аргументы функции». Необходимо задать массив в котором находится матрица ($E - A$), Вводим массив E6:F7 (рисунок 6).

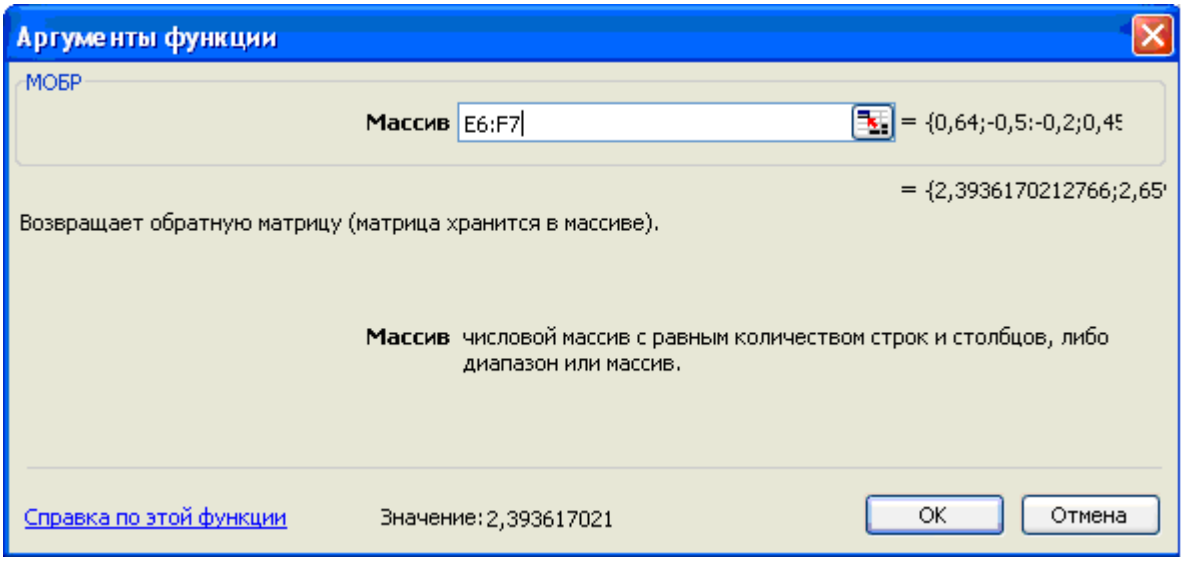
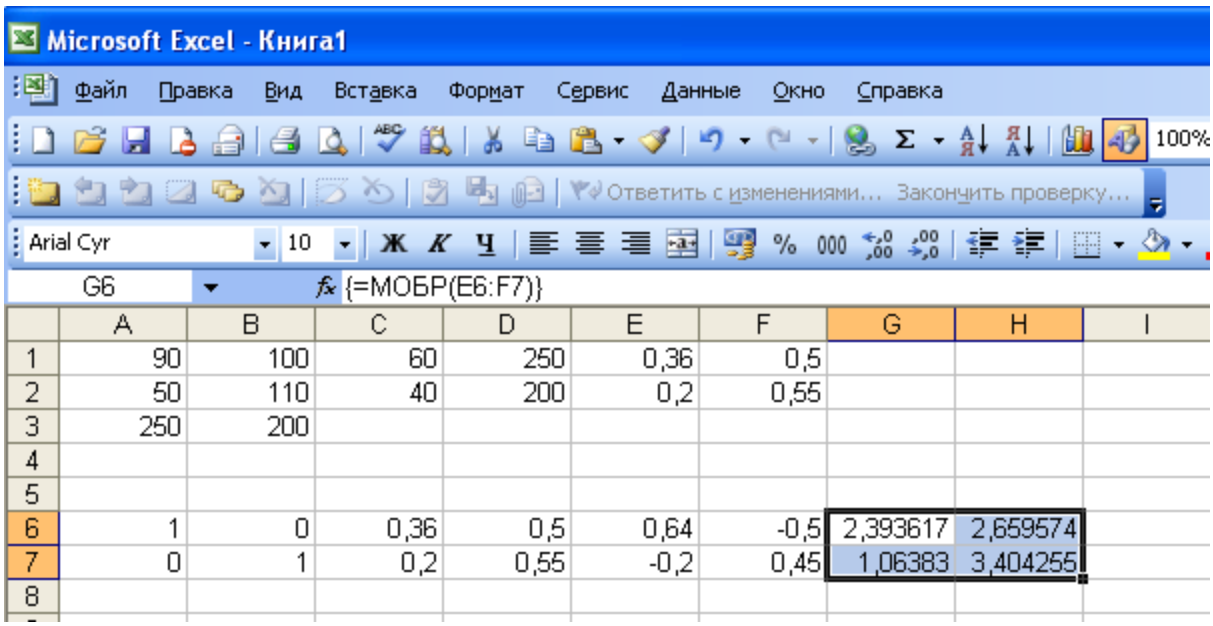


Рисунок 6 – Ввод данных при расчете обратной матрицы

Для отображения результата в виде матрицы, нажмите Shift+Ctrl-Enter (если нажать ОК, то в ячейке G6 будет одно число). Массив G6:H7 будет содержать искомую матрицу $B = (E - A)^{-1}$ (рисунок 7).



	A	B	C	D	E	F	G	H	I
1	90	100	60	250	0,36	0,5			
2	50	110	40	200	0,2	0,55			
3	250	200							
4									
5									
6	1	0	0,36	0,5	0,64	-0,5	2,393617	2,659574	
7	0	1	0,2	0,55	-0,2	0,45	1,06383	3,404255	
8									

Рисунок 7 – Результат расчета обратной матрицы

В качестве проверки можно рассчитать матрицу *X*, Матрица *X* рассчитывается по формуле $X = BY$. Введем в диапазон I6:I7 матрицу *Y*. Выделим диапазон J6:J7, выберем команду «Вставка → Функция...». В открывшемся окне «Мастер функций» выберем категорию «Математические» и из них МУМНОЖ (рисунок 8).

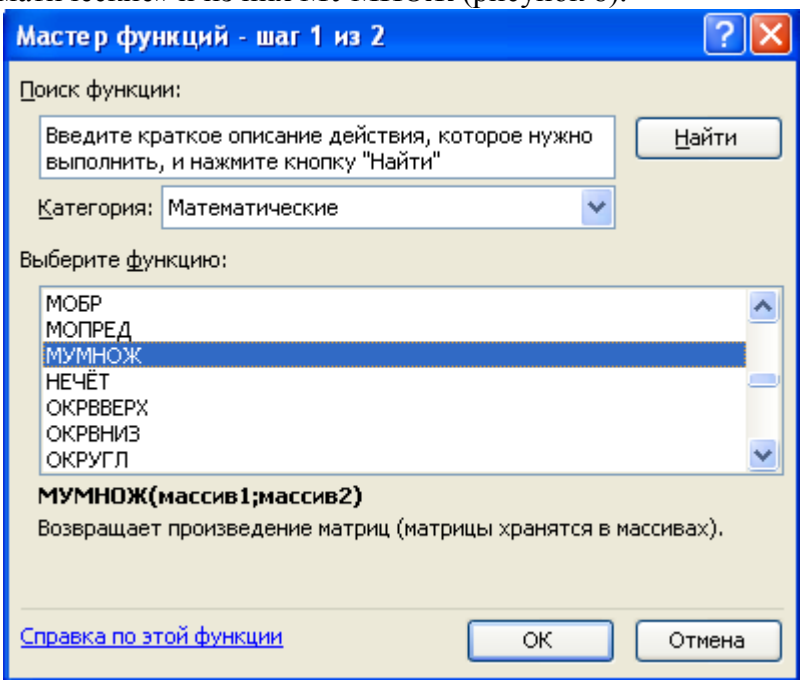


Рисунок 8 – Окно «Мастер функций»

Нажмите ОК. Откроется окно «Аргументы функции». Необходимо указать массивы, в которых находятся перемножаемые матрицы (порядок ввода массивов имеет значение), в нашем примере это массивы G6:H7 и I6:I7 (рисунок 9).

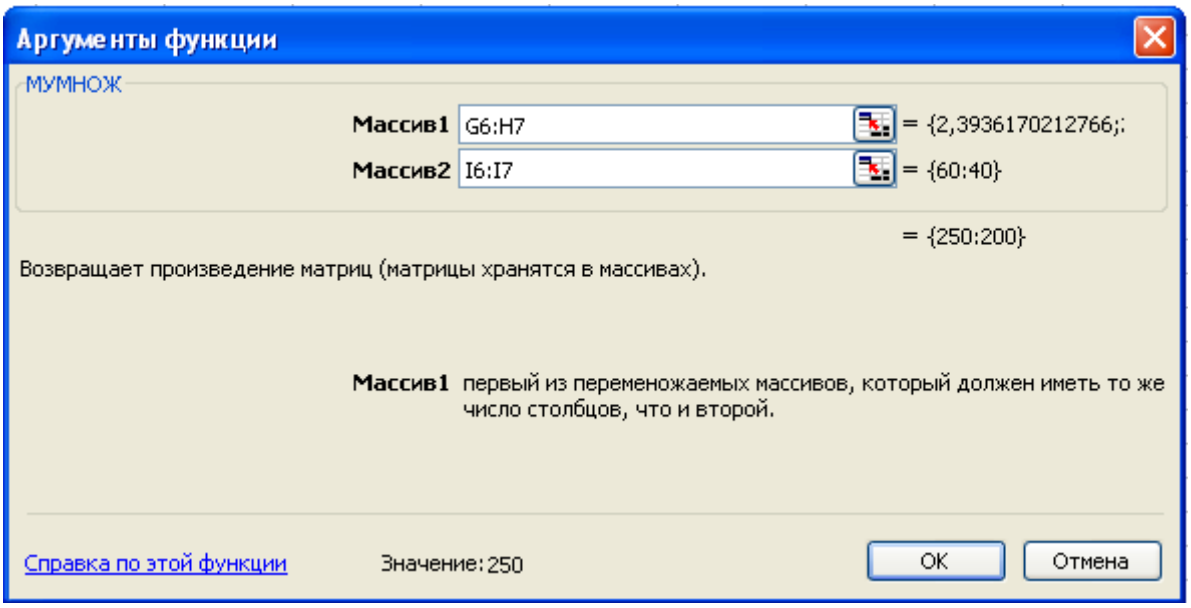
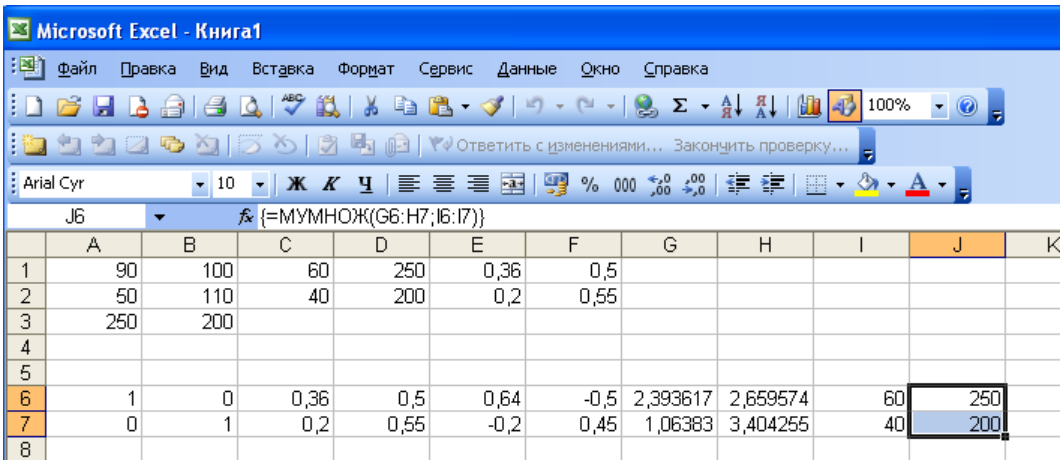


Рисунок 9 – Ввод данных при перемножении матриц

После окончания ввода данных нажмите Shift+Ctrl+Enter. Массив J6:J7 будет содержать искомую матрицу X (рисунок 10).



	A	B	C	D	E	F	G	H	I	J	K
1	90	100	60	250	0,36	0,5					
2	50	110	40	200	0,2	0,55					
3	250	200									
4											
5											
6	1	0	0,36	0,5	0,64	-0,5	2,393617	2,659574	60	250	
7	0	1	0,2	0,55	-0,2	0,45	1,06383	3,404255	40	200	
8											

Рисунок 10 – Результат расчета матрицы X

Ответ: Объемы производства валовой продукции равны $X_1 = 250$; $X_2 = 200$; коэффициенты прямых затрат равны

$$A = \begin{pmatrix} 0,36 & 0,5 \\ 0,2 & 0,55 \end{pmatrix}$$

коэффициенты полных материальных затрат равны

$$B = \begin{pmatrix} 2,39 & 2,66 \\ 1,06 & 3,40 \end{pmatrix}$$

Задача 7

Используя коэффициенты прямых материальных затрат, представленные в таблице 9 и объемы конечного продукта по отраслям рассчитать полные материальные затраты и объемы производства валовой продукции.

Таблица 9 – Исходные данные к задаче 7

Производящие отрасли	Потребляющие отрасли		Конечный продукт
	1	2	
1	0,36	0,15	70
2	0,40	0,25	50

Задача 8

На основании данных, приведенных в нижеследующих таблицах (10, 11, 12), рассчитать коэффициенты прямых и полных материальных затрат.

Таблица 10 – Исходные данные к задаче 8 (А)

Потребляющие отрасли	Производящие отрасли			Конечный продукт
	1	2	3	
1	50	60	80	60
2	25	90	40	105
3	25	60	40	85

Таблица 11 – Исходные данные к задаче 8 (Б)

Потребляющие отрасли	Производящие отрасли			Конечный продукт
	1	2	3	
1	40	18	25	71
2	16	9	25	36
3	40	45	50	115

Таблица 12 – Исходные данные к задаче 8 (В)

Потребляющие отрасли	Производящие отрасли			Конечный продукт
	1	2	3	
1	18	36	25	61
2	45	90	25	20
3	36	36	50	30

Задача 9

В таблице 13 приведены коэффициенты прямых материальных затрат и объемы конечной продукции в межотраслевом балансе для трех отраслей.

Таблица 13 – Исходные данные к задаче 9

Отрасль	Коэффициенты прямых затрат			Конечный продукт
	1	2	3	
1	0,2	0,2	0,1	50
2	0,4	0,3	0,2	40
3	0,2	0,2	0,4	30

Требуется рассчитать коэффициенты полных материальных затрат и найти объемы валовой продукции отраслей.

Задача 10

В таблице 14 приведены коэффициенты прямых материальных затрат и объемы конечной продукции в межотраслевом балансе для трех отраслей.

Таблица 14 – Исходные данные к задаче 10

Отрасль	Коэффициенты прямых затрат			Конечный продукт
	1	2	3	
1	0,3	0,4	0,2	40
2	0,2	0,1	0,3	15
3	0,1	0,2	0,2	10

Требуется рассчитать коэффициенты полных материальных затрат и найти объемы валовой продукции отраслей.

Задача 11

На основе данных задачи 9 восстановите схему межотраслевого материального баланса.

Задача 12

На основе данных задачи 10 восстановите схему межотраслевого материального баланса.

Задача 13

Используя данные баланса (таблица), определите объемы производства валовой продукции, коэффициенты прямых и полных материальных затрат.

Таблица 15 – Исходные данные к задаче 13

Производящие отрасли	Потребляющие отрасли		Конечный продукт
	1	2	
1	10	17	23
2	20	15	35

Задача 14

Используя данные баланса (таблица 16), определите объемы производства валовой продукции, коэффициенты прямых и полных материальных затрат.

Таблица 3.16 – Исходные данные к задаче 14

Производящие отрасли	Потребляющие отрасли		Конечный продукт
	1	2	
1	70	45	25
2	25	30	40

Задача 15

Используя коэффициенты прямых материальных затрат, представленных в таблице 17 и объемы конечного продукта по отраслям рассчитать полные материальные затраты и объемы производства валовой продукции.

Таблица 17 – Исходные данные к задаче 15

Производящие отрасли	Потребляющие отрасли		Конечный продукт
	1	2	
1	0,22	0,54	20
2	0,38	0,26	17

К числу важнейших аналитических возможностей балансового метода относится определение прямых и полных затрат труда на единицу продукции и разработка на этой основе балансовых продуктово-трудовых моделей, исходной моделью при этом служит отчетный межпродуктовый баланс в натуральном выражении. В этом балансе по строкам представлено распределение каждого отдельного продукта на производство других продуктов и конечное потребление (первый и второй квадранты схемы межотраслевого баланса). Отдельной строкой дается распределение затрат живого труда в производстве всех видов продукции. Предполагается, что трудовые затраты выражены в единицах труда одинаковой степени сложности.

Обозначим затраты живого труда в производстве *j*-го продукта через L_j, а объем производства этого продукта (валовой выпуск), как и раньше, через X_j. Тогда, прямые затраты труда на единицу *j*-го вида продукции (*коэффициент прямой трудоемкости*) можно задать следующей формулой:

$$t_j = \frac{L_j}{X_j}; j = \overline{1,n}. \quad (12)$$

Из данной формулы следует, что

$$L_j = X_j\, t_j. \quad (13)$$

Если межотраслевые прямые затраты труда обозначить через l_{ij}, то они будут соответственно равны:

$$l_{ij} = x_{ij}\, t_i. \quad (14)$$

Введем понятие *прямых затрат труда* как суммы прямых затрат живого труда и затрат овеществленного труда, перенесенных на продукт через израсходованные средства производства. Если обозначить величину полных затрат труда на единицу продукции *j*-го вида через T_j, то произведения вида $a_{ij}T_j$ отражают затраты овеществленного труда, перенесенного на единицу *j*-го продукта через *i*-ое средство производства; при этом предполагается, что коэффициенты прямых материальных затрат a_{ij} выражены в натуральных единицах. Тогда полные трудовые затраты на единицу *j*-го вида продукции (*коэффициент полной трудоемкости*) будут равны:

$$T_j = \sum_{i=1}^{n} a_{ij}T_i + t_j; j = \overline{1,n}. \quad (15)$$

Введем в рассмотрение вектор-строку коэффициентов прямой трудоемкости $t = (t_1, t_2,\ldots, t_n)$ и вектор строку коэффициентов полной трудоемкости $T = (T_1, T_2,\ldots, T_n)$.

Тогда, с использованием уже рассматриваемой выше матрицы коэффициентов прямых материальных затрат *A* (в натуральном выражении) систему уравнений (15) можно переписать в матричном виде:

$$T = T\,A + t. \quad (16)$$

Произведя очевидные матричные преобразования с использованием единичной матрицы *E*

$$T - TA = TE - TA = T\,(E - A) = t,$$

получим следующее соотношение для вектора коэффициентов полной трудоемкости:

$$T = t\,(E - A)^{-1}. \quad (17)$$

Матрица $(E - A)^{-1}$, это матрица *B* коэффициентов полных материальных затрат, так что последнее равенство можно переписать в виде:

$$T = t\,B. \quad (17')$$

Обозначим через *L* величину совокупных затрат живого труда по всем видам продукции, которая с учетом формулы (12) будет равна:

$$L = \sum_{j=1}^{n} L_j = \sum_{j=1}^{n} t_j X_j = tX. \quad (18)$$

Используя соотношения (18), приходим к следующему неравенству:

$$tX = TY, \quad (19)$$

где t и T – вектор-строки коэффициентов прямой валовой и конечной продукции соответственно.

На основе коэффициентов прямой и полной трудоемкости могут быть разработаны межотраслевые и межпродуктовые балансы затрат труда и использования трудовых ресурсов. Схематически эти балансы строятся по общему типу матричных моделей.

Развитие основной модели межотраслевого баланса достигается также путем включения в нее показателей фондоемкости продукции. В простейшем случае модель дополняется отдельной строкой, в которой указаны в стоимостном выражении объемы производственных фондов Φ_j, занятые в каждой j-ой отрасли. На основании этих данных и объемов валовой продукции всех отраслей определяются коэффициенты прямой фондоемкости продукции j-ой отрасли:

$$f_j = \frac{\Phi_j}{X_j}; \; j = \overline{1,n}. \tag{20}$$

Стоимость основных производственных фондов, занятых в каждой j-ой отрасли соответственно равна:

$$\Phi_j = X_j f_j. \tag{21}$$

Стоимость производственных фондов j-ой отрасли, занятых при производстве продукции для i-ой отрасли будет равна:

$$\phi_{ij} = x_{ij} f_j \; . \tag{22}$$

Коэффициент прямой фондоемкости показывает величину производственных фондов, непосредственно занятых в производстве данной отрасли, в расчете на единицу ее валовой продукции. В отличии от этого показателя коэффициент полной фондоемкости F_j отражает объем фондов, необходимых во всех отраслях для выпуска единицы конечной продукции j-ой отрасли. Если a_{ij} – коэффициент прямых материальных затрат, то для коэффициента полной фондоемкости справедливо равенство, аналогичное равенству (13) для коэффициента полной трудоемкости:

$$F_j = \sum_{i=1}^{n} a_{ij} F_i + f_j; \; j = \overline{1,n}. \tag{23}$$

Если ввести в рассмотрение вектор-строку коэффициентов прямой фондоемкости $f = (f_1, f_2, \ldots, f_n)$ и вектор-строку коэффициентов полной фондоемкости $F = (F_1, F_2, \ldots, F_n)$, то систему уравнений (23) можно переписать в матричной форме:

$$F = F A + f, \tag{24}$$

откуда с помощью преобразований, аналогичных применяемым выше для коэффициентов трудоемкости, можно получить матричное соотношение:

$$F = f B, \tag{25}$$

где $B = (E - A)^{-1}$ – матрица коэффициентов полных материальных затрат.

Задача 16

Межотраслевой баланс производства и распределения продукции представлен в таблице 18.

Таблица 18 – Исходные данные к задаче 16

Производящие отрасли	Потребляющие отрасли			Конечная продукция	Валовая продукция
	1	2	3		
1	232,6	51,0	291,8	200,0	775,3
2	155,1	255,0	0,0	100,0	510,1
3	232,6	51,0	145,9	300,0	729,6
Условно чистая продукция	155,0	153,1	291,9		
Валовая продукция	775,3	510,0	729,6		

Заданы затраты живого труда (трудовые ресурсы) в трех отраслях: $L_1 = 1160$, $L_2 = 460$, $L_3 = 875$ в некоторых единицах измерения трудовых затрат. Требуется определить коэффициенты прямой и полной трудоемкости и составить межотраслевой баланс затрат труда.

Решение

1) Находим коэффициенты прямой трудоемкости

$$t_j = \frac{L_j}{X_j},$$

$$t_1 = \frac{1160}{775{,}3} = 1{,}5; \qquad t_2 = \frac{460}{510{,}1} = 0{,}9; \qquad t_3 = \frac{875}{729{,}6} = 1{,}2.$$

2) Рассчитаем матрицу коэффициентов полных материальных затрат

а) Вычислим коэффициенты прямых затрат (a_{ij}) по формуле:

$$a_{ij} = \frac{x_{ij}}{X_j},\ i = 1, \ldots, n,\ j = 1, \ldots, n.$$

$a_{11} = 232{,}6 : 775{,}3 = 0{,}3$; $a_{12} = 51{,}0 : 510{,}1 = 0{,}1$; $a_{13} = 291{,}8 : 729{,}6 = 0{,}4$;
$a_{21} = 155{,}1 : 775{,}3 = 0{,}2$; $a_{22} = 255{,}0 : 510{,}1 = 0{,}5$; $a_{23} = 0{,}0 : 729{,}6 = 0$;
$a_{31} = 232{,}6 : 775{,}3 = 0{,}3$; $a_{32} = 51{,}0 : 510{,}1 = 0{,}1$; $a_{33} = 145{,}9 : 729{,}6 = 0{,}2$.

Рассчитаем матрицу B:

$$B = (E - A)^{-1} = \begin{pmatrix} 2{,}041 & 0{,}612 & 1{,}020 \\ 0{,}816 & 2{,}245 & 0{,}408 \\ 0{,}867 & 0{,}510 & 1{,}684 \end{pmatrix}.$$

Находим коэффициенты полной трудоемкости:

$$T = tB \Rightarrow \quad T = (1{,}5;0{,}9;1{,}2) \cdot \begin{pmatrix} 2{,}041 & 0{,}612 & 1{,}020 \\ 0{,}816 & 2{,}245 & 0{,}408 \\ 0{,}867 & 0{,}510 & 1{,}684 \end{pmatrix} = (4{,}84;3{,}55;3{,}92).$$

Умножая первую, вторую и третью строки первого и второго квадрантов межотраслевого материального баланса на соответствующие коэффициенты прямой трудоемкости, получаем схему межотраслевого баланса труда (в трудовых измерителях):

$$l_{ij} = x_{ij} t_i,$$

$l_{11} = 232{,}6 \cdot 1{,}5 = 348{,}9$, $l_{12} = 51{,}0 \cdot 1{,}5 = 76{,}5$, $l_{13} = 291{,}8 \cdot 1{,}5 = 437{,}7$,
$l_{y1} = 200 \cdot 1{,}5 = 300$,
$l_{21} = 155{,}1 \cdot 0{,}9 = 139{,}6$, $l_{22} = 255{,}0 \cdot 0{,}9 = 229{,}5$, $l_{23} = 0{,}0 \cdot 0{,}9 = 0$,
$l_{y2} = 100 \cdot 0{,}9 = 90$,
$l_{31} = 232{,}6 \cdot 1{,}2 = 279{,}1$, $l_{32} = 51{,}0 \cdot 1{,}2 = 61{,}2$, $l_{33} = 145{,}9 \cdot 1{,}2 = 175{,}1$,
$l_{y3} = 300 \cdot 1{,}2 = 360$.

Ответ:

Межотраслевой баланс затрат труда

Таблица 19 – Итоговая матрица задачи 16

Отрасль	Межотраслевые затраты овеществленного труда			Затраты труда на конечную продукцию	Затраты труда в отраслях
	1	2	3		
1	348,9	76,5	437,7	300,0	1163,0
2	139,6	229,5	0,0	90,0	459,1
3	279,1	61,2	175,1	360,0	875,5

Замечание: незначительные расхождения между полученными данными в таблице 19 и исходными данными по затратам живого труда вызваны погрешностью округления при вычислении.

Задача 17

По данным межотраслевого баланса, представленного в таблице 20 и затратам живого труда $L_1 = 80$, $L_2 = 45$, $L_3 = 90$, определить коэффициенты прямой и полной трудоемкости.

Таблица 20 – Исходные данные к задаче 17

Производящие отрасли	Потребляющие отрасли			Конечная продукция	Валовая продукция
	1	2	3		
1	18	7	5	21	51
2	6	8	2	20	36
3	3	15	14	23	55

Задача 18

По данным межотраслевого баланса, представленного в таблице 21 и затратам живого труда $L_1 = 300$, $L_2 = 290$, $L_3 = 450$, определить коэффициенты прямой и полной трудоемкости.

Таблица 21 – Исходные данные к задаче 18

Производящие отрасли	Потребляющие отрасли			Конечная продукция	Валовая продукция
	1	2	3		
1	90	56	64	240	450
2	45	85	210	310	650
3	83	98	101	518	800

Задача 19

По данным межотраслевого баланса, представленного в таблице 22 и стоимости производственных фондов $\Phi_1 = 1250$, $\Phi_2 = 1700$, $\Phi_3 = 1010$, определить коэффициенты прямой и полной фондоемкости.

Таблица 22 – Исходные данные к задаче19

Производящие отрасли	Потребляющие отрасли			Конечная продукция	Валовая продукция
	1	2	3		
1	180	210	115	405	748
2	250	80	170	620	1120
3	112	87	35	276	510

Задача 20

По данным межотраслевого баланса, представленного в таблице 23 и стоимости производственных фондов $\Phi_1 = 83$, $\Phi_2 = 58$, $\Phi_3 = 75$, определить коэффициенты прямой и полной фондоемкости.

Таблица 23 – Исходные данные к задаче 20

Производящие отрасли	Потребляющие отрасли			Конечная продукция	Валовая продукция
	1	2	3		
1	9	5	6	37	57
2	4	7	1	23	35
3	11	8	6	45	70

Задача 21

По данным схемы межотраслевого баланса, представленного в таблице 24, и затрат труда $L_1 = 2950$, $L_2 = 3100$, $L_3 = 1500$, составить схему межотраслевого баланса труда.

Таблица 24 – Исходные данные к задаче 21

Производящие отрасли	Потребляющие отрасли			Конечная продукция	Валовая продукция
	1	2	3		
1	830	715	390	1980	3915
2	650	817	235	1200	2902
3	350	185	148	737	1420

Задача 22

По данным схемы межотраслевого баланса, представленного в таблице 25 и затрат труда $L_1 = 100$, $L_2 = 102$, $L_3 = 163$, составить схему межотраслевого баланса труда.

Таблица 25 – Исходные данные к задаче 22

Производящие отрасли	Потребляющие отрасли			Конечная продукция	Валовая продукция
	1	2	3		
1	15	22	12	31	80
2	17	13	23	15	68
3	35	15	10	37	97

Задача 23

По данным схемы межотраслевого баланса, представленного в таблице 26 и стоимости производственных фондов каждой из отраслей $\Phi_1 = 1053$, $\Phi_2 = 1200$, $\Phi_3 = 3090$, составить схему межотраслевого баланса производственных фондов.

Таблица 26 – Исходные данные к задаче 23

Производящие отрасли	Потребляющие отрасли			Конечная продукция	Валовая продукция
	1	2	3		
1	250	345	127	682	1404
2	101	485	320	809	1715
3	713	305	513	1044	2575

Задача 24

По данным схемы межотраслевого баланса, представленного в таблице 27 и стоимости производственных фондов каждой из отраслей $\Phi_1 = 809$, $\Phi_2 = 673$, $\Phi_3 = 1005$, составить схему межотраслевого баланса производственных фондов.

Таблица 27 – Исходные данные к задаче 24

Производящие отрасли	Потребляющие отрасли			Конечная продукция	Валовая продукция
	1	2	3		
1	310	218	415	790	1733
2	98	170	53	315	636
3	436	275	119	710	1540

3. Контрольная работа.

2.4 Лабораторная работа №12, 13, 14, 15 (8 часов).

Тема: «Экономические задачи, решаемые с применением корреляционно-регрессионного анализа и организация статистического моделирования с применением программы Statistica»

2.4.1 Цель работы: изучить виды экономических задач, решаемые с применением корреляционно-регрессионного анализа и организация статистического моделирования с применением программы Statistica»

2.4.2 Задачи работы:

1. Изучение видов экономических задач, решаемые с применением корреляционно-регрессионного анализа

2. Использование при организации статистического моделирования программного продукта Statistica

2.4.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. OpenOffice

2. Microsoft Office Excel,

3. ППП Statistica 6.0.

2.4.4 Описание (ход) работы:

1. Постановка задачи. Ввод данных в ППП Statistica 6.0.

Различают два типа связей между различными явлениями и их признаками: *функциональную,* или *жестко детерминированную,* с одной стороны, и *статистическую,* или *стохастически детерминированную*, – с другой. Строго определить различие этих типов связи можно тогда, когда они получают математическую формулировку. Для простоты будем говорить о связи двух явлений или двух признаков, математически отображаемой в форме уравнения связи двух переменных.

Если с изменением значения одной из переменных вторая изменяется строго определенным образом, т.е. значению одной переменной обязательно соответствует одно или несколько точно заданных значений другой переменной, связь между ними является *функциональной.*

Нередко говорят о строгом соответствии лишь одного значения второй из переменных каждому значению первой из них, но это неверно. Например, связь между y и x является строго функциональной, если $y = 4x$; но значению $x = 4$ соответствует не одно, а два значения: $y_1 = +2$, $y_2 = -2$. Уравнения более высоких степеней могут иметь несколько корней, связь, разумеется остается функциональной.

Функциональная связь двух величин возможна лишь при условии, что вторая из них зависит только от первой. В реальной природе (и тем более в обществе) таких связей нет; они являются лишь абстракциями, полезными и необходимыми при анализе явлений, но упрощающими реальность. Функциональная зависимость данной величины y от многих факторов $X_1, X_2, ..., X_n$ возможна только в том случае, если величина y всегда зависит только от перечисленного набора факторов $X_1, X_2, ..., X_n$ и ни от чего более. Все явления и процессы реального мира связаны между собой, и нет такого конечного числа переменных k, которое абсолютно полно определяло бы собой зависимую величину y. Следовательно, множественная функциональная зависимость переменных есть тоже абстракция, упрощающая реальность.

Однако, механика, электротехника, акустика, политическая экономия и другие науки успешно используют представление связей как функциональных не только в аналитических целях, но нередко и в целях прогнозирования. Это возможно потому, что в простых системах интересующая нас переменная величина зависит в основном (скажем, на 99% или даже на 99,99%) от немногих других переменных или только от одной переменной, т.е. связь является хотя и не абсолютно функциональной, но практически очень близкой к таковой.

Стохастически детерминированная связь не имеет ограничений и условий, присущих функциональной связи. Если с изменением значения одной из переменных вторая может в определенных пределах принимать любые значения с некоторыми вероятностями, но ее среднее значение или иные статистические (массовые) характеристики изменяются по определенному закону, связь является статистической. Иными словами, при статистической связи разным значениям одной переменной соответствуют разные распределения значений другой переменной.

В настоящее время наука не знает более широкого определения связи. Все связи, которые могут быть измерены и выражены численно, подходят под определение «статистические связи», в том числе и функциональные. Последние представляют собой частный случай статистических связей, когда значениям одной переменной соответствуют «распределения» значений второй, состоящие из одного или нескольких значений и имеющие вероятность, равную единице. Конечно, качественное различие действительно вероятностных распределений и отдельных значений, имеющих вероятность единицы (достоверных), настолько велико, что хотя функциональные связи и могут рассматриваться как предельный случай статистической связи, все же с полным основанием можно говорить о двух типах связей.

Корреляционной связью называют важнейший частный случай статистической связи, состоящий в том, что разным значениям одной переменной соответствуют различные средние значения другой. С изменением значения признака x закономерным образом изменяется среднее значение признака y, в то время как в каждом отдельном случае значение признака y (с различными вероятностями) может принимать множество различных значений.

Если же с изменением значения признака x среднее значение признака y не изменяется закономерным образом, но закономерно изменяется другая статистическая характеристика (показатели вариации, асимметрии, эксцесса и т.п.), то связь не является корреляционной, но статистической.

Статистическая связь между двумя признаками (переменными величинами) предполагает, что каждый из них имеет случайную вариацию индивидуальных значений относительно средней величины. Если же такую вариацию имеет только один из признаков, а значения другого являются жестко детерминированными, то говорят лишь о регрессии. Например, при анализе динамических рядов можно измерять регрессию уровней ряда урожайности (имеющих случайную колеблемость) на номера лет. Но нельзя говорить о корреляции между ними и применять показатели корреляции с соответствующей интерпретацией.

Корреляционная связь между признаками может возникнуть разными путями. Первый (важнейший) путь – *причинная зависимость* результативного признака (его вариации) от вариации факторного признака. Например, признак x – балл оценки плодородия почв, признак y – урожайность сельскохозяйственной культуры. Здесь совершенно ясно логически, какой признак выступает как независимая переменная (фактор x), какой – как зависимая переменная (результат y).

Второй путь – *сопряженность,* возникающая при наличии общей причины. Известен классический пример: если в качестве признака x взять число пожарных команд в городе, а за признак y – сумму убытков за год в городе от пожаров, то между признаками x и y в совокупности городов России существовала прямая корреляция: в среднем чем больше пожарников в городе, тем больше и убытков от пожаров. Возникает вопрос – не сами ли пожарники занимались поджогами, тем самым создавая себе работу. Но дело в другом. Данную корреляцию нельзя

интерпретировать как связь причины и следствия; оба признака – следствия общей причины – размера города. Вполне логично, что в крупных городах больше пожарных частей, но больше и пожаров, и убытков от них за год, чем в малых городах.

Третий путь возникновения корреляции – *взаимосвязь признаков,* каждый из которых и причина, и следствие. Такова, например, корреляция между уровнями производительности труда рабочих и уровнем оплаты 1 ч труда (тарифной ставкой). С одной стороны, уровень зарплаты – следствие производительности труда: чем она выше, тем выше и оплата. Но, с другой стороны, установленные тарифные ставки и расценки играют стимулирующую роль: при правильной системе оплаты они выступают в качестве фактора, от которого зависит производительность труда. В такой системе признаков допустимы обе постановки задачи; каждый признак может выступать в роли независимой переменной x и в качестве зависимой переменной y.

Парная линейная корреляция. Простейшей системой корреляционной связи является линейная связь между двумя признаками – *парная линейная корреляция.*

Практическое ее значение в том, что есть системы, в которых среди всех факторов, влияющих на результативный признак, выделяется один важнейший фактор, который в основном определяет вариацию результативного признака. Измерение парных корреляций составляет необходимый этап в изучении сложных, многофакторных связей. Есть и такие системы связей, при изучении которых следует предпочесть парную корреляцию. Внимание к линейным связям объясняется ограниченной вариацией переменных и тем, что в большинстве случаев нелинейные формы связей для выполнения расчетов преобразуются в линейную форму (линеаризуются).

Уравнение парной линейной корреляционной связи называется *уравнением парной регрессии* и имеет вид:

$$y = a + bx, \qquad (1)$$

где y – среднее значение результативного признака y при определенном значении факторного признака x;

a – свободный член уравнения;

b – коэффициент регрессии, измеряющий среднее отношение отклонения результативного признака от его средней величины к отклонению факторного признака от его средней величины на одну единицу его измерения, – вариация y, приходящаяся на единицу вариации x.

Линейные связи являются основными. Однако встречаются и нелинейные связи, хорошо описываемые параболой, гиперболой и т.д.

Параболическая корреляция. Уравнение регрессии в форме параболы 2-го порядка имеет следующий вид:

$$y = a + bx + cx^2. \qquad (2)$$

Если при линейной связи среднее изменение результативного признака на единицу фактора постоянно по всей области вариации фактора, то при параболической корреляции изменение признака x на единицу признака y меняется равномерно с изменением величины фактора. В результате связь может даже поменять знак на противоположный, из прямой превратится в обратную, из обратной в прямую. Такой характер связи объективно присущ многим системам. Например, с увеличением дозы удобрений урожайность сельскохозяйственных культур сначала повышается, но если превысить оптимальную величину дозы, то при дальнейшем росте дозы удобрений растения угнетаются и урожайность снижается.

Гиперболическая корреляция. Уравнение регрессии в форме гиперболы имеет следующий вид:

$$y = a + b/x. \qquad (3)$$

Если величина b положительна, то при увеличении значений факторного признака x значения результативного признака уменьшаются, причем это уменьшение все время замедляется, и при $x \to \infty$ средняя величина признака y будет равна a. Если же параметр b

отрицателен, то значения результативного признака с ростом фактора возрастают, причем их рост замедляется, и в пределе при $x \to \infty$, $y = a$. Таким образом, гиперболические зависимости характерны для связей, в которых результативный признак не может варьировать неограниченно, его вариация имеет односторонний предел. Например, при освоении нового оборудования его производительность возрастает, но рост замедлится по мере приближения к конструктивно-технологическому пределу производственной мощности агрегата. Совершенствуя двигатель, можно увеличивать его КПД, но тоже не выше предела, допустимого данным видом преобразования энергии. Таков же характер связи между уровнем душевого дохода x в семье и долей семей, имеющих телевизоры, y; он приближен к пределу (100%) в наиболее обеспеченной группе семей.

Показатели тесноты связи. Для практического использования моделей регрессии большое значение имеет их адекватность, т.е. соответствие фактическим статистическим данным.

Корреляционный и регрессионный анализ обычно (особенно в условиях так называемого малого и среднего бизнеса) проводится для ограниченной по объёму совокупности. Поэтому показатели регрессии и корреляции – параметры уравнения регрессии, коэффициенты корреляции и детерминации могут быть искажены действием случайных факторов. Чтобы проверить, насколько эти показатели характерны для всей генеральной совокупности, необходимо проверить адекватность построенных статистических моделей.

При численности объектов анализа до 30 единиц возникает необходимость проверки значимости (существенности) каждого коэффициента регрессии. При этом выясняют, насколько вычисленные параметры характерны для отображения комплекса условий: не являются ли полученные значения параметров результатами действия случайных или сторонних причин.

Значимость коэффициентов простой линейной регрессии (применительно к совокупностям, у которых $n < 30$) осуществляют с помощью t-критерия Стьюдента. При этом вычисляют расчетные (фактические) значения t-критерия для параметров уравнения.

Вычисленные по вышеприведенным формулам значения сравнивают их с критическими t, которые определяют по таблице Стьюдента с учетом принятого уровня значимости a и числом степеней свободы вариации $v = n - 2$. В социально-экономических исследованиях уровень значимости a обычно принимают равным 0,05. Параметр признаётся значимым (существенным) при условии, если $t_{расч} > t_{табл}$. В таком случае практически невероятно, что найденные значения параметров обусловлены только случайными совпадениями.

Проверка адекватности регрессионной модели может быть дополнена корреляционным анализом. Для этого необходимо определить *тесноту* корреляционной связи между переменными x и k. Теснота корреляционной связи, как и любой другой, может быть измерена *эмпирическим корреляционным отношением* k, когда n (межгрупповая дисперсия) характеризует отклонения групповых средних результативного признака от общей средней.

Говоря о корреляционном отношении как о показателе измерения тесноты зависимости, следует отличать от эмпирического корреляционного отношения – *теоретическое.*

Теоретическое корреляционное отношение n представляет собой относительную величину, получающуюся в результате сравнения среднего квадратического отклонения выровненных значений результативного признака δ, то есть рассчитанных по уравнению регрессии, со средним квадратическим отношением эмпирических (фактических) значений результативности признака σ. Изменение значения n объясняется влиянием факторного признака.

Множественная корреляция. Выше рассматривалась зависимость между двумя признаками, то есть речь шла о парной корреляции. На практике же чаще всего изменение изучаемого признака зависит от нескольких причин. В таких случаях изучение

корреляционной связи не может ограничиться парными зависимостями, и в анализ необходимо включить другие признаки – факторы, существенно влияющие на изучаемую зависимую переменную.

Если обозначить факторы через x_1, x_2, x_3... x_m, то линейное уравнение множественной зависимости может быть записано следующим образом:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \ldots + b_mx_m. \quad (4)$$

По параметрам уравнения можно оценить долю каждого из факторов в изменении уровня результативного показателя y. Это может быть сделано путем прямой оценки по величине коэффициентов регрессии при каждом из факторов, а также по коэффициентам эластичности и стандартизированным частным коэффициентам регрессии.

При построении многофакторных корреляционных моделей одной из предпосылок обоснованности конечных результатов является требование наименьшей коррелированности включенных в модель признаков-факторов (отсутствие мультиколлинеарности). В случае наличия линейной зависимости между факторами система нормальных уравнений не будет иметь однозначного решения, в результате чего коэффициенты регрессии и другие оценки окажутся неустойчивыми.

Изучение как парной, так и множественной корреляции наиболее удобно осуществлять в специализированных статистических программных продуктах, каким и является программный комплекс Statistica. Данный программный комплекс предназначен для статистического анализа данных в среде Windows. Кроме парной и множественной корреляции, Statistica реализует другие углубленные методы анализа данных на компьютере в области экономики, маркетинга, рекламы, бизнеса и других сфер.

Рассмотрим основы работы в программном комплексе Statistica 6.0 и, в частности, его применение для решения задач корреляционно-регрессионного моделирования.

После запуска программы Statistica появится Рабочее Окно системы (рисунок 1):

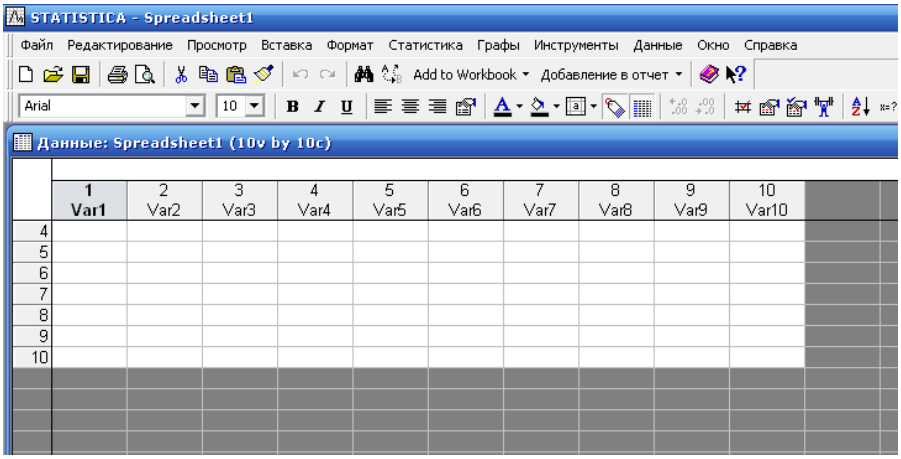


Рисунок 1 – Рабочее Окно системы Statistica

Рабочее окно системы Statistica имеет следующую структуру: верхний заголовок Statistica: Basic Statistics and Tables (Основные статистики и таблицы), – запущен модуль Basic Statistics and Tables, (если бы был запущен другой модуль, то его название указывалось бы в заголовке); строка меню, панель инструментов и рабочая область, занимающая большую часть окна. В рабочую область выводятся все документы системы, которые получаются в процессе анализа.

В строке заголовка находятся три кнопки, позволяющие изменять размеры окна:

Кнопка минимизации размеров окна. Окно модуля будет свернуто до размеров кнопки на Панели задач Windows.

Кнопка восстановления размеров окна. Она появляется, если Statistica запущена в полноэкранном режиме (основное окно занимает весь экран целиком). Если щелкнуть на ней, то размер окна уменьшится, а вид самой кнопки измени[тся] на кнопку максимизации

размера окна, при помощи которой вновь можно развернуть окно до размеров полного экрана.

КНОПКА ЗАКРЫТИЯ ОКНА. ЩЕЛЧОК НА НЕЙ ПРИВЕДЕТ К ЗАКРЫТИЮ ОКНА И ВЫХОДУ ИЗ ОТКРЫТОГО МОДУЛЯ.

Меню занимает вторую строку основного окна модуля и содержит в себе систему выпадающих меню. Ряд пунктов меню: **Файл, Правка, Вид, Окно, Справка,** стандартен для Windows, пункт **Анализ** специфичен для Statistica.

Панель инструментов содержит кнопки для быстрого доступа к командам меню. При помощи щелчка мышью на какой-либо кнопке можно получить быстрый доступ к соответствующей команде. Каждому типу документа Statistica соответствует своя панель инструментов. Внешний вид панели инструментов и ее расположение в окне системы можно настроить при помощи команды **Toolbar (Панель инструментов)** из меню **View (Вид)**. Эти установки действуют для текущего сеанса работы. Панель инструментов может быть выведена в одну и две строчки и может быть расположена в разных частях основного окна системы. Постоянный вид панели инструментов установлен в меню **Options (Опции)** командой **Display (Экран)**.

При следующих запусках автоматически открывается последний файл. В рабочей области может находится только один файл с **исходными данными** .

Исходные данные в системе Statistica организованы в виде электронной таблицы, где они имеют разные смысловые значения.

Столбцы электронной таблицы с исходными данными называются ***Variables (Переменные),*** а строки ***Cases (Случаи)***. В качестве переменных обычно выступают исследуемые величины, а случаи – это значения, которые принимают переменные в отдельных измерениях.

Система может работать как с численными, так и с *текстовыми данными*, это важно в практических статистических исследованиях. Также, они поддерживают различные типы операций с использованием *буфера обмена Windows;* операции с выделенными блоками значений, в том числе и с использованием метода *drag-and-drop,* автозаполнение блоков и т.д.

Открытие файла данных

Подведите курсор мыши к пункту меню **Файл** и щелкните левой кнопкой. В выпадающем меню, которое появилось на экране, выберите **Открытие,** либо нажмите сочетание клавиш Ctrl+O на клавиатуре**.**

Выберите в каталоге stat\examples файл cars.sta, как показано на рисунок 2. Нажмите кнопку **OK.**

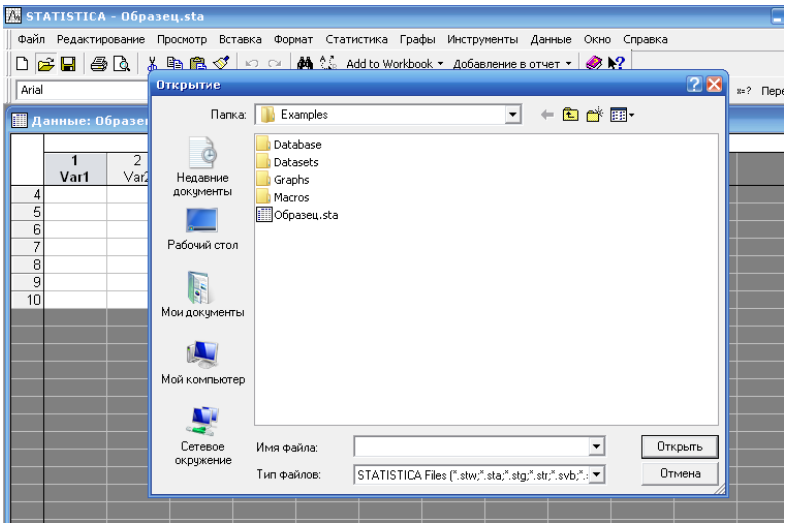


Рисунок 2 – Открытие файла образец.sta из каталога stat\examples

2. Решение задач в ППП Statistica 6.0. Анализ решения и экономическая интерпретация.

Примеры создания файлов данных

Исходное положение: вы находитесь в основном рабочем окне системы Statistica.

Начальные действия: подведите курсор мыши к строке меню к пункту **Файл** и щелкните левой кнопкой.

В верхней части окна расположена строка меню. В выпадающем меню, появившемся на экране после щелчка мыши, выберите команду **НОВЫЙ.**

Пример создания файла *Реклама 1.STA*

В таблице 1 приведены данные о размере рекламного объявления и его цене при размещении на полосе газеты.

Создадим файл **Statistica** с этими данными.

Таблица 1 – Исходные данные

Длина (мм)	Ширина (мм)	Площадь (мм)	Цена (руб.)
47	35	1645	1446000
47	73	3431	2768000
47	111	5217	3974000
47	149	7003	5147000
47	209	9823	6290000
47	225	10575	7537000
47	263	12361	882800
47	301	14147	10230000

Шаг 1. Создание электронной таблицы. Выберите команду **НОВЫЙ** из меню **Файл.** В появившемся диалоговом окне **Создание нового документа** переставьте галочку меню **Размещение** на позицию **«В новой рабочей книге»** (рисунок 3).

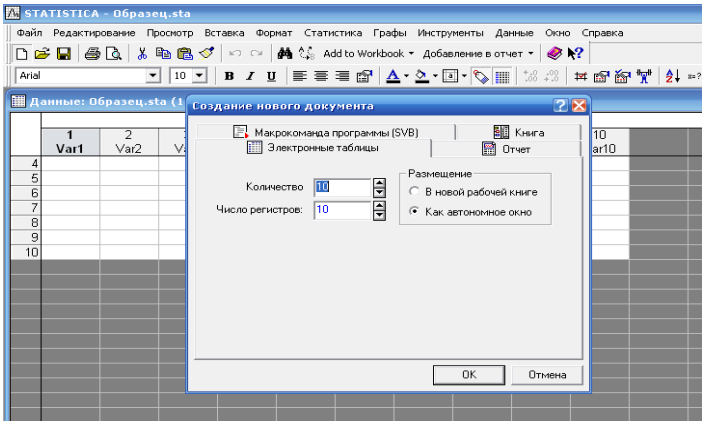


Рисунок 3 – Создание нового файла

Нажмите кнопку **ОК** в правом нижнем углу окна. **Statistica** автоматически откроет пустую электронную таблицу с именем Spreadsheet#, в новой рабочей книге Workbook# которая и появится на экране (рисунок 4)

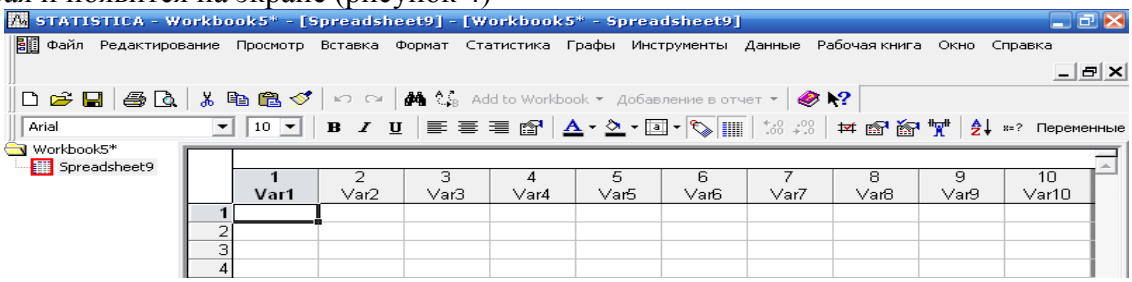


Рисунок 4 – Пустая электронная таблица для ввода данных

Вы можете пользоваться этой таблицей как страницей в записной книжке и внести в нее необходимые вам данные.

В заголовке окна электронной таблицы автоматически отображается имя файла Spreadsheet# и название рабочей книги Workbook#.

Размер таблицы по умолчанию принят 10 на 10 (10 переменных с именами VAR1, VAR2, VAR3… VAR10 и 10 пронумерованных строк).

Сделаем в таблице столько строк и столбцов, сколько нужно. Нам нужно, чтобы в таблице имелось 4 переменные и 8 случаев.

Задайте имя таблице, нажав меню **ФАЙЛ/ Сохранить,** далее появится следующее меню рисунок 5).

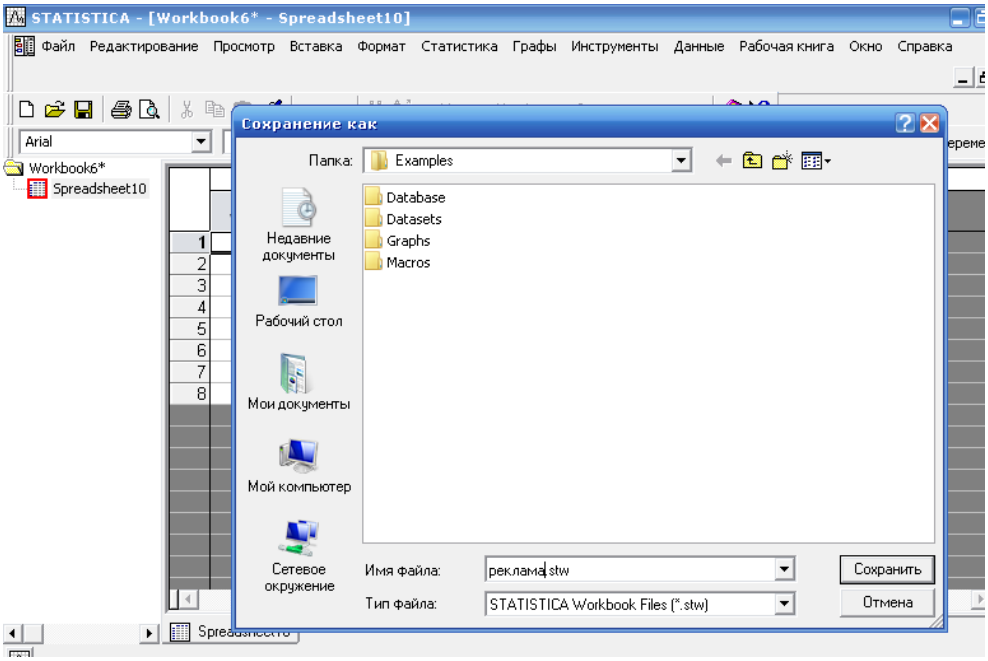


Рисунок 5 – Задание имени таблицы

В поле **имя файла** введите «Реклама» и затем переименуйте рабочий лист (рисунок 6).

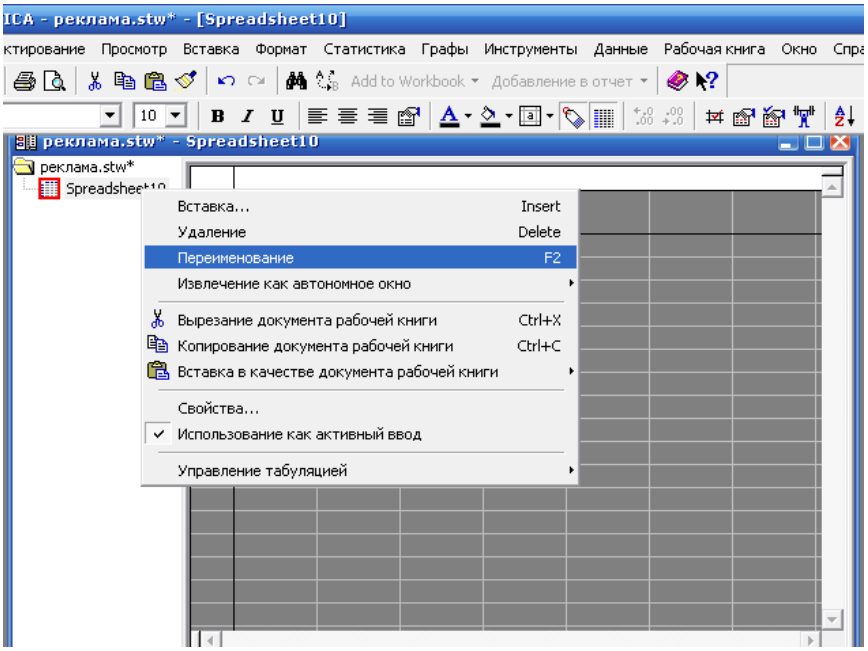


Рисунок 6 – Переименование листа

В появившемся поле ввода запишите реклама 1.

Шаг 2. Настройка таблицы. Произведем настройку размеров таблицы. Для данных рекламного объявления требуется четыре переменных: ширина рекламы, длина, площадь, цена.

Нажмите кнопку **Переменные** на панели инструментов и выберите команду **Удалить.** В диалоговом окне **Delete Variables (Удаление переменных)** укажите диапазон удаляемых переменных, как показано на рисунках 7. и 8. Нажмите кнопку **ОК**.

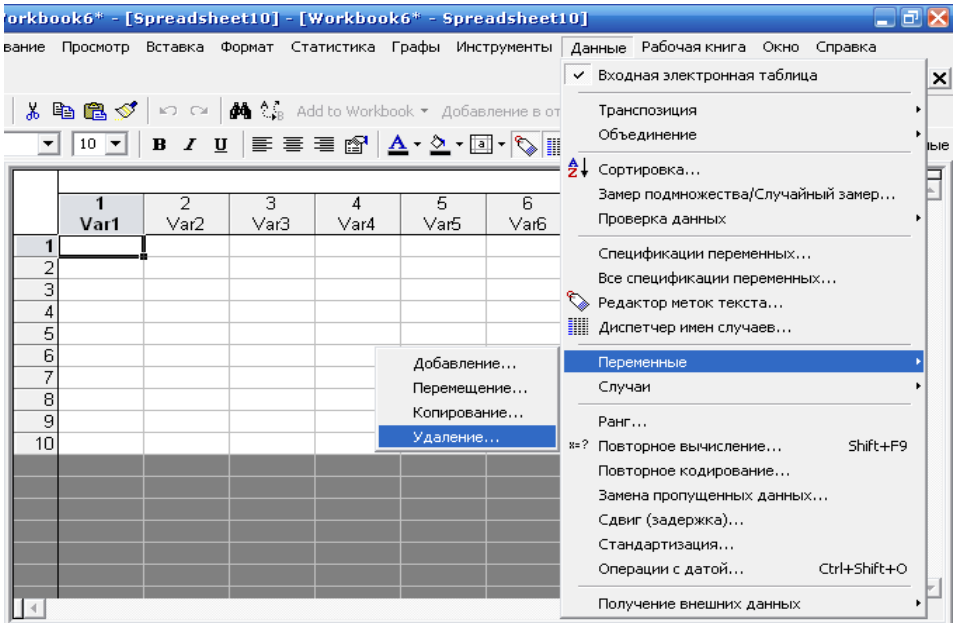


Рисунок 7 – Окно удаления ненужных переменных

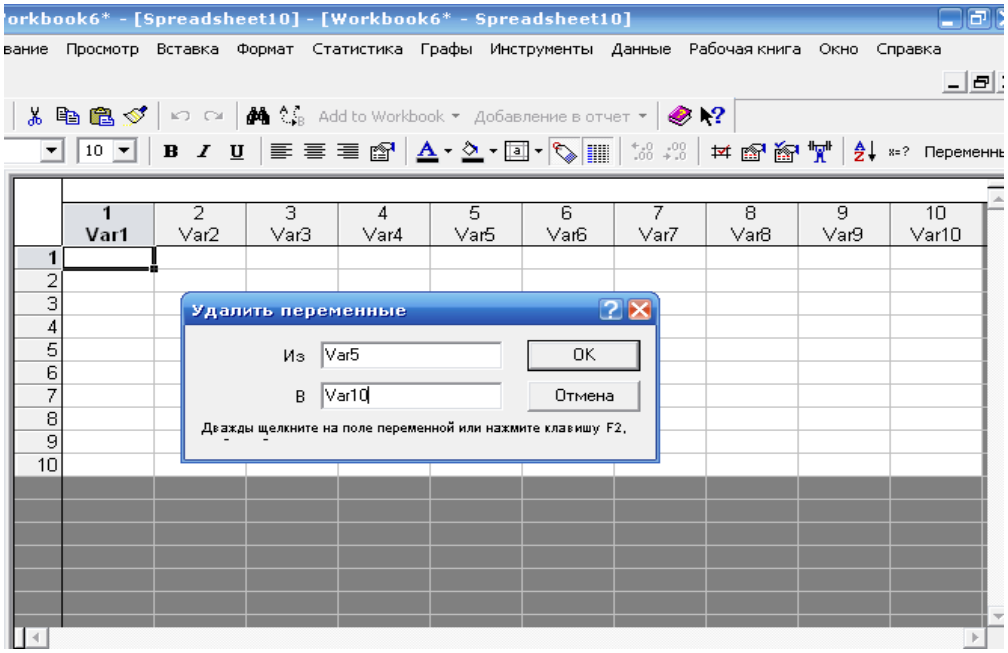


Рисунок 8 – Окно удаления ненужных переменных (продолжение)

Необходимое число случаев – 8. В созданной таблице число случаев равно 10. Два лишних случая из таблицы следует удалить.

Для удаления лишних случаев аналогичным способом происходит удаление случаев (рисунок 9).

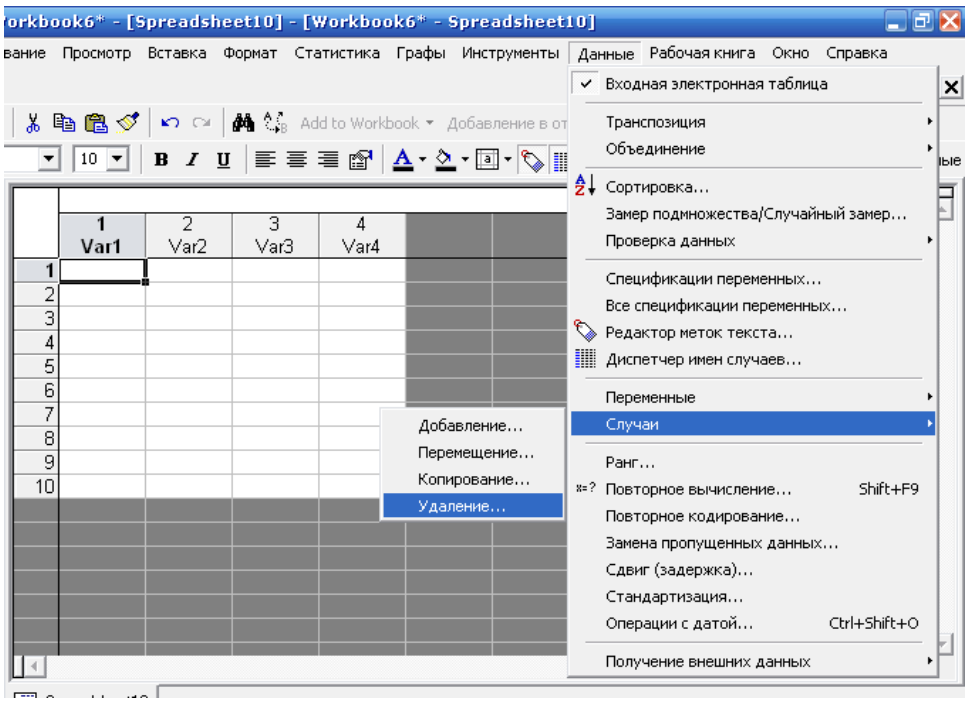


Рисунок 9 – Окно удаления лишних случаев

Задайте диапазон удаляемых случаев в диалоговом окне **Удалить случаи**. Нажмите кнопку **ОК**. Теперь электронная таблица выглядит согласно рисунка 10.

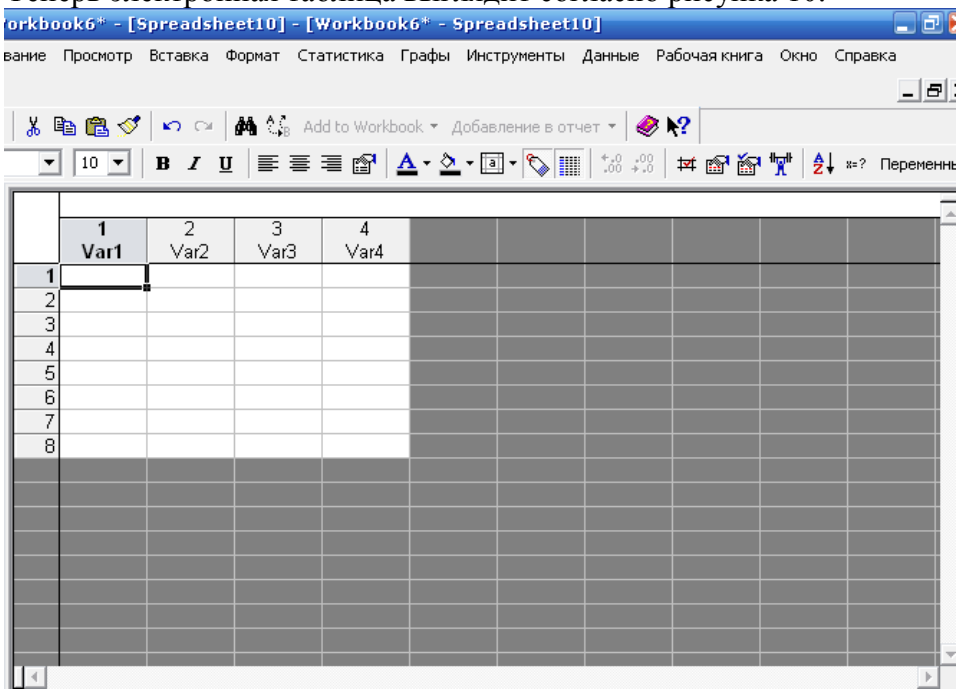


Рисунок 10 – Пустая таблица с четырьмя переменными и восемью случаями

Шаг 3. Подготовка таблицы к вводу данных, заголовок таблицы и имена переменных. Дважды щелкните мышью на белом поле в таблице под словами: Data: REKLAMAL.STA4V*8C.

На экране появится окно **Data File Header (Заголовок файла данных)**, в котором можно задать заголовок таблицы и дополнительную информацию о данных. Введем заголовок таблицы: ЦЕНА РЕКЛАМЫ, теперь таблица выглядит как показано на рисунке 11.

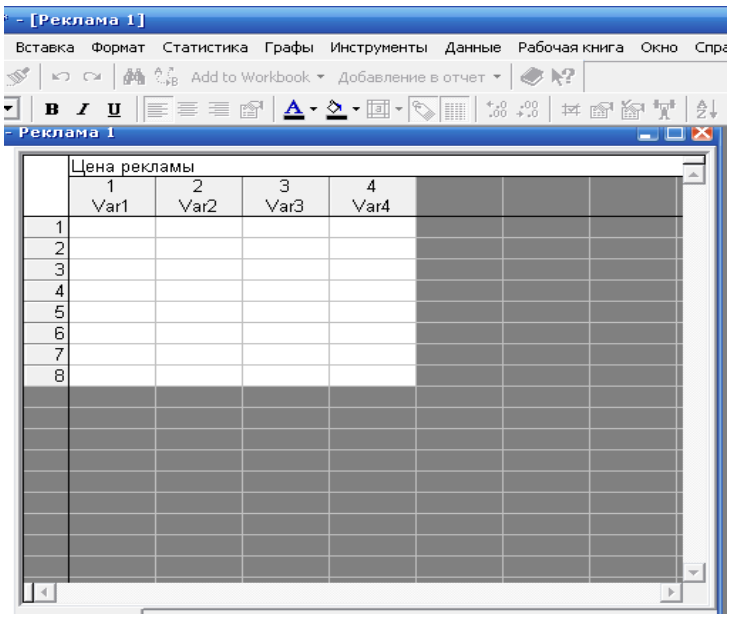


Рисунок 11 – Таблица с заголовком

Таблица почти готова к вводу данных, но придадим ей еще более удобный вид: введем имена переменных, которые отражают смысл записей, и специфируем их.

Задайте имена переменных.

Дважды щелкните на имени переменной VAR1 в электронной таблице. На экране появится окно спецификации переменной VAR1 (рисунок 12). В поле **Name (Имя)** напишите: ЦЕНА. Длина имени не должна превышать 8 символов.

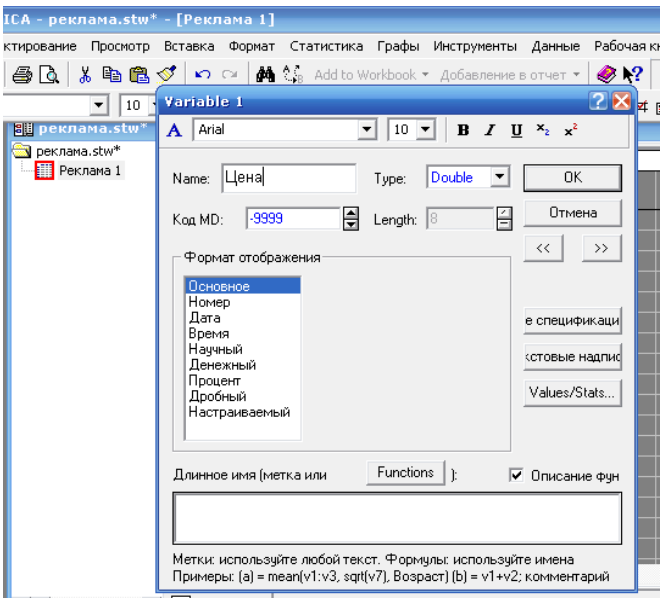


Рисунок 12 – Окно спецификации переменной VAR1

Нажмите кнопку **ОК**.

То же сделайте для переменной VAR2, ей присвойте имя ДЛИНА, переменной VAR3 присвойте имя ПЛОЩАДЬ. Переменной VAR4 присвойте имя ЦЕНА (рисунок 13).

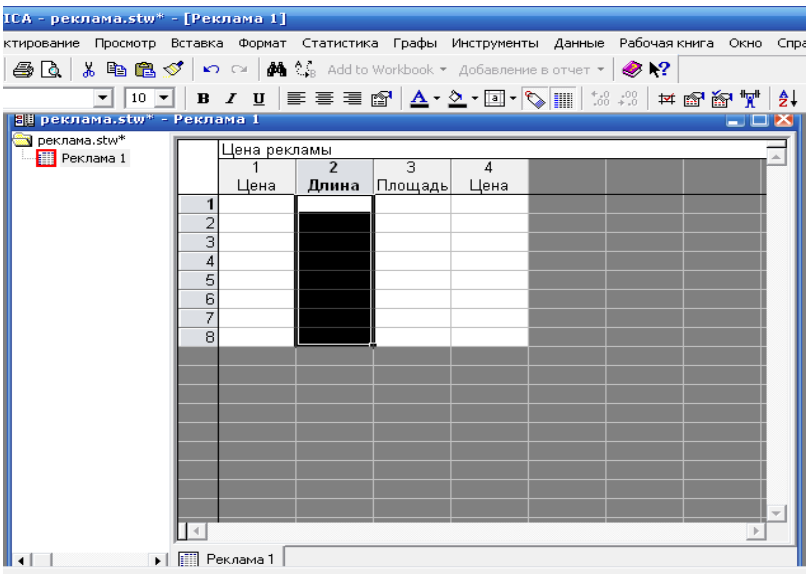


Рисунок 13 – Таблица с новыми именами переменных

Теперь таблица готова к тому, чтобы ввести в нее данные.

Шаг 4. Ввод данных в электронную таблицу. Введите данные, как показано на рисунке 14.

Data: NEW.STA 4v * 8c

ЦЕНА РЕКЛАМЫ

	1 ШИРИНА	2 ДЛИНА	3 ПЛОЩАДЬ	4 ЦЕНА
1	47	35		1446000
2	47	73		2768000
3	47	111		3974000
4	47	149		5147000
5	47	187		6290000
6	47	225		7537000
7	47	263		8828000
8	47	301		10260000

Рисунок 14 – Таблица с введенными с клавиатуры данными

Данные вводятся только в колонки ДЛИНА, ШИРИНА, ЦЕНА.

Колонка ПЛОЩАДЬ остается пустой.

Введем данные в колонку ПЛОЩАДЬ. Сделаем это прямым подсчетом площади прямоугольника по длине и ширине.

Дважды щелкните на имени переменной ПЛОЩАДЬ в электронной таблице. На экране появится окно спецификации переменной ПЛОЩАДЬ (рисунок 15.).

В поле **Long names (Длинные имена)** запишите формулу =v1*v2 (рисунок 15). Площадь прямоугольника равна произведению длины на ширину. Нажмите кнопку **ОК.**

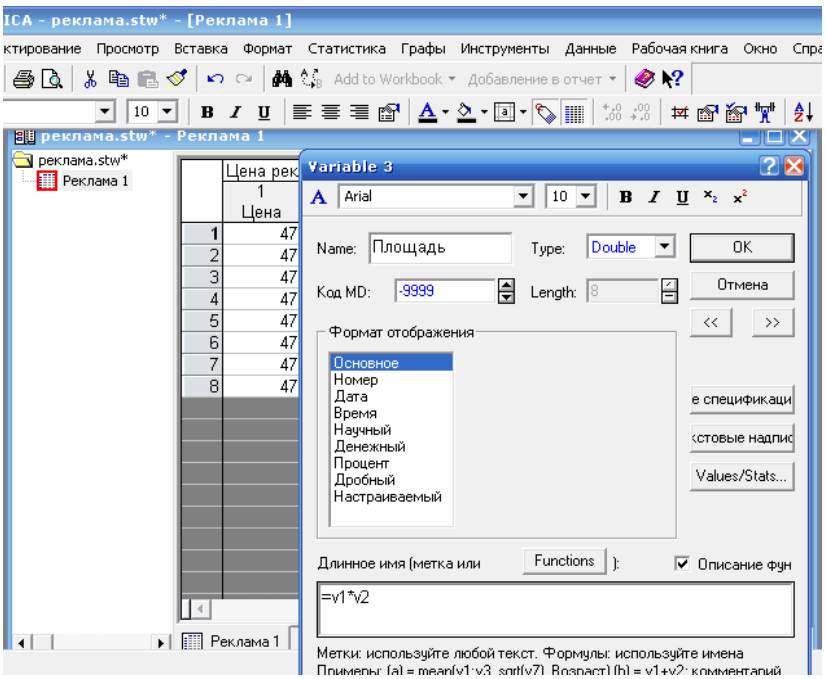


Рисунок 15 – Вычисление значений переменной ПЛОЩАДЬ

Площадь рекламных объявлений будет подсчитана и занесена в ячейки переменной VAR3. Полностью заполненная таблица стоимости рекламных объявлений на полосе газет появится на вашем экране (рисунок 16).

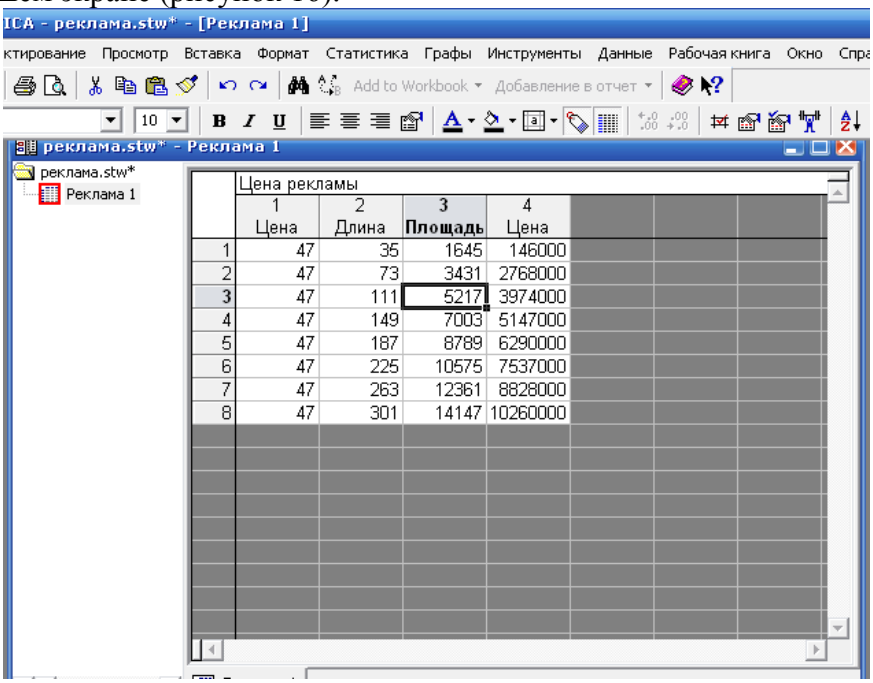


Рисунок 16 – Полностью заполненная электронная таблица с исходными данными

Шаг 5. Сохранение файла данных. Для сохранения созданного файла нажмите мышью на кнопку **Save Data File (Сохранить файл данных)**.

Задания

Задача 1

Предприятие на протяжении длительного времени выпускает носочные изделия. Запланировав модернизацию производства, руководство предприятия решило провести исследования, цель которых состоит в выявлении зависимости цены носков от ряда приведенных факторов, таких, как плотность, состав и фирма-производитель. Цена носков –

это зависимая переменная *Y*. В качестве независимых, объясняющих переменных были выбраны:

- плотность. Обозначим через x_1;
- содержание шерсти. Обозначим через x_2;
- содержание хлопка. Обозначим через x_3;
- фирма-производитель. Обозначим через x_4.

Для решения задачи имеются исходные данные, представленные в приложении 1, которые необходимо занести в программный комплекс **Statistica**. Для в главном меню выбираем команду Файл – Новый (рисунок 17).

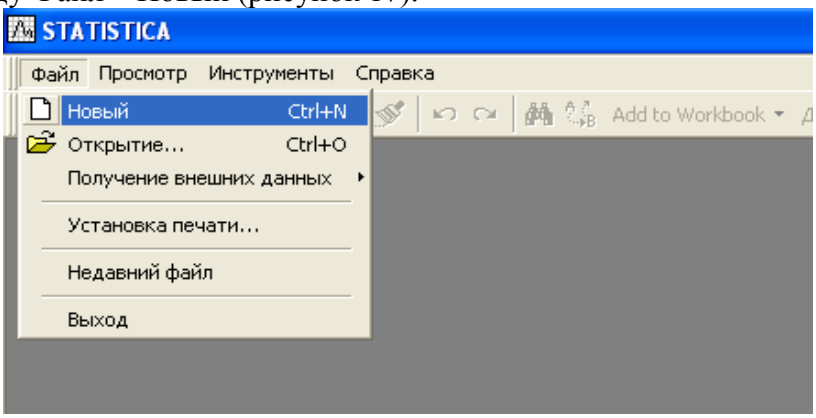


Рисунок 17 – Открытие нового файла

Откроется диалог для создания новой таблицы (рисунок 18).

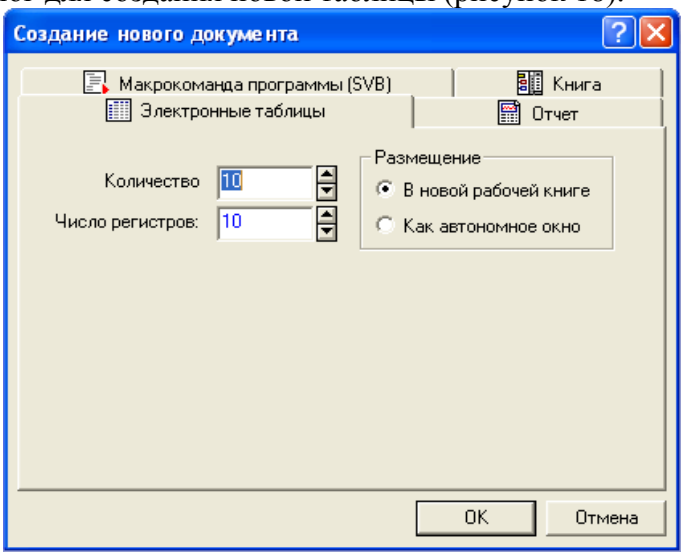
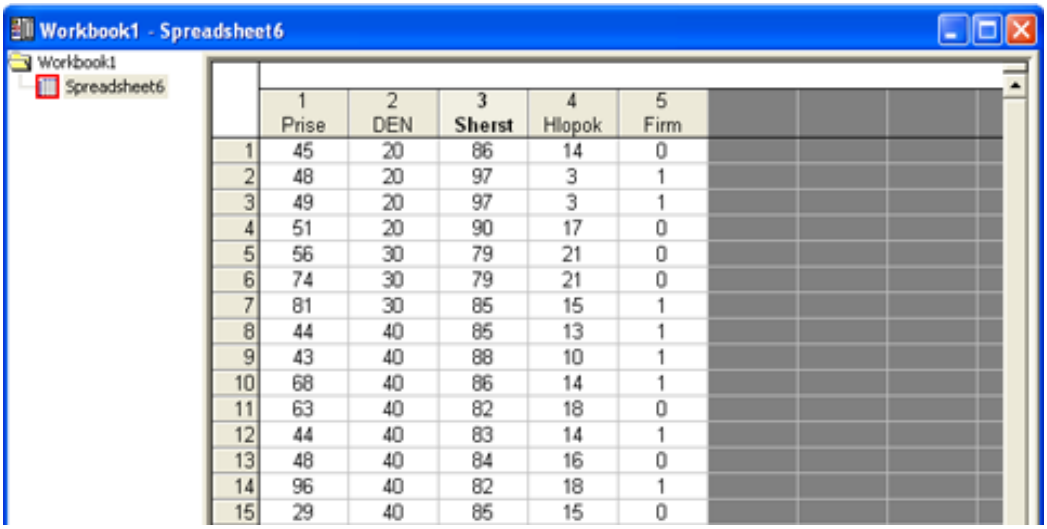


Рисунок 4.18 – Создание нового документа

Нужно изменить значения параметров «Количество», указывающее на число факторов и «Число регистров», указывающее на их количество. В случае нашей задачи получится 5 на 45.

После создания таблицы занесем в нее данные (рисунок 19).



	1 Prise	2 DEN	3 Sherst	4 Hlopok	5 Firm
1	45	20	86	14	0
2	48	20	97	3	1
3	49	20	97	3	1
4	51	20	90	17	0
5	56	30	79	21	0
6	74	30	79	21	0
7	81	30	85	15	1
8	44	40	85	13	1
9	43	40	88	10	1
10	68	40	86	14	1
11	63	40	82	18	0
12	44	40	83	14	1
13	48	40	84	16	0
14	96	40	82	18	1
15	29	40	85	15	0

Рисунок 19 – Занесение данных

Для удобства можно переименовать названия столбцов. Для этого два раза кликнем на заголовке столбца и в открывшимся диалоге изменяем значение поля Name.

Сначала покажем *парную корреляцию* результативного фактора y и одного из образующих факторов. Для примера возьмем зависимость цены от плотности носок (x_2).

Для этого нужно в главном меню выбрать Статистика – Множественная регрессия (рисунок 20).

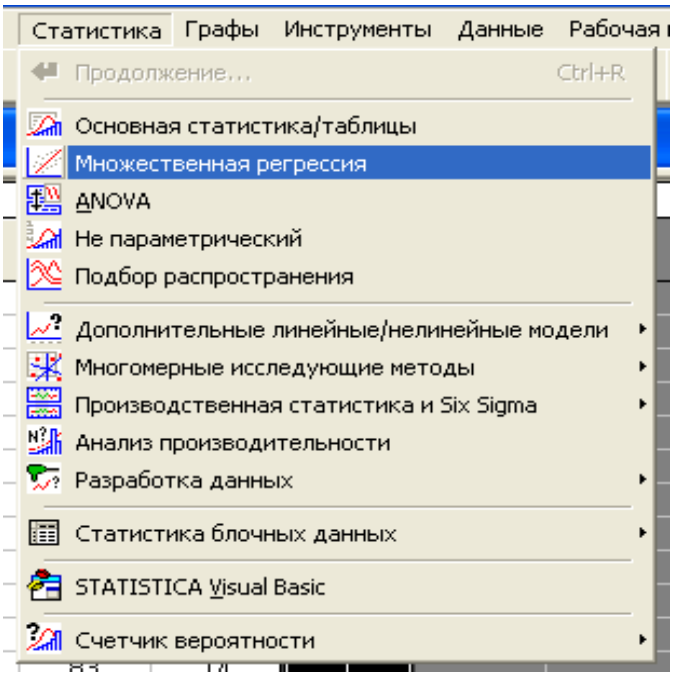


Рисунок 20 – Активизация множественной регрессии

В открывшемся диалоговом окне нажимаем кнопку Variables и выбираем в качестве зависимой переменной Price (первый столбец), а в качестве независимой – показатель плотности DEN (второй столбец) (рисунок 21).

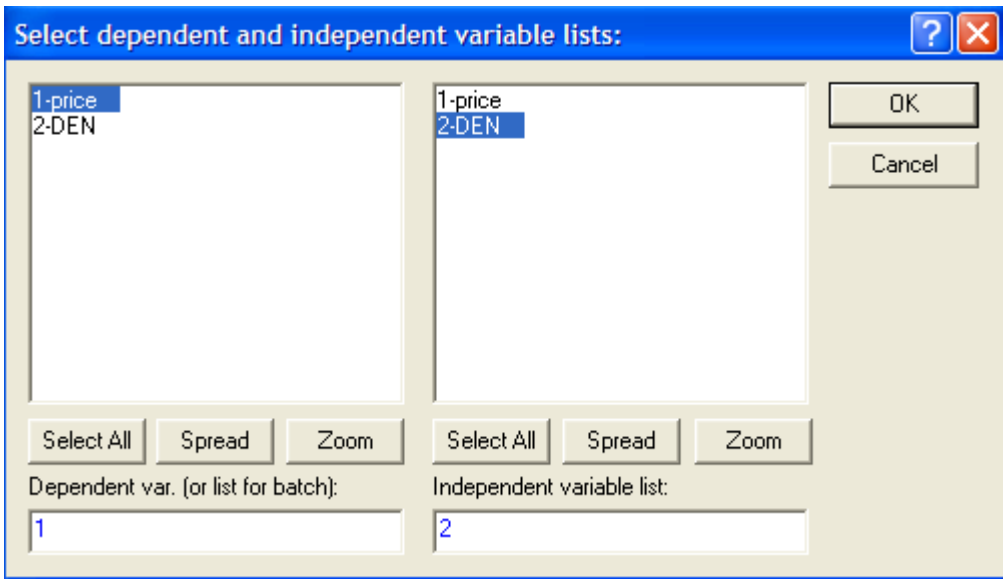


Рисунок 21 – Выбор зависимой и независимой переменных

Нажимаем кнопку OK. В вернувшемся окне также нажимаем OK, ничего не меняя в параметрах (рисунок 22).

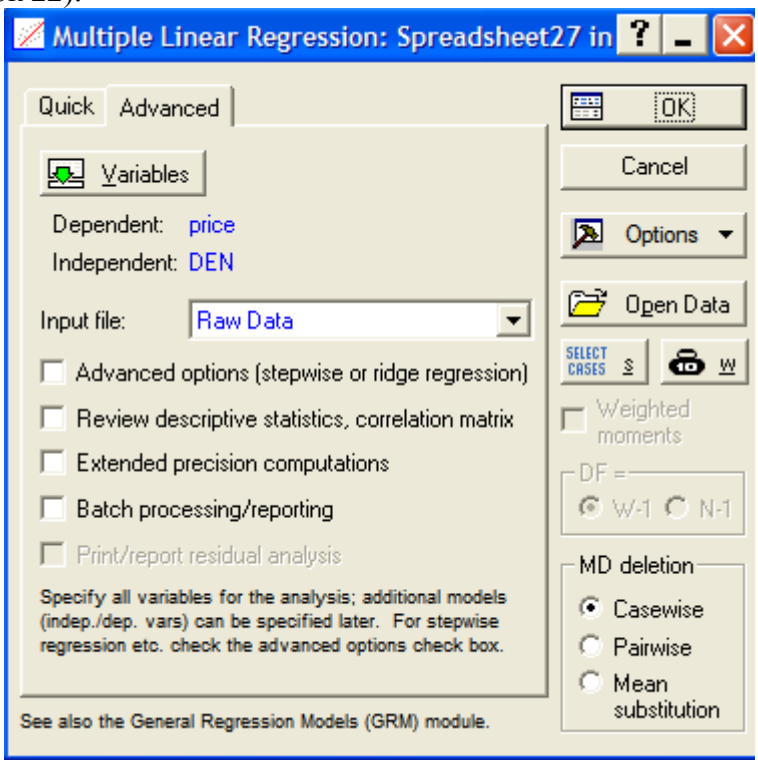


Рисунок 22 – Линейная регрессии

В результате получаем отчет, из которого нам необходимо получить таблицу с коэффициентами. Для этого нажимаем кнопку Summary: Regressions Results внизу окна (рисунок 23).

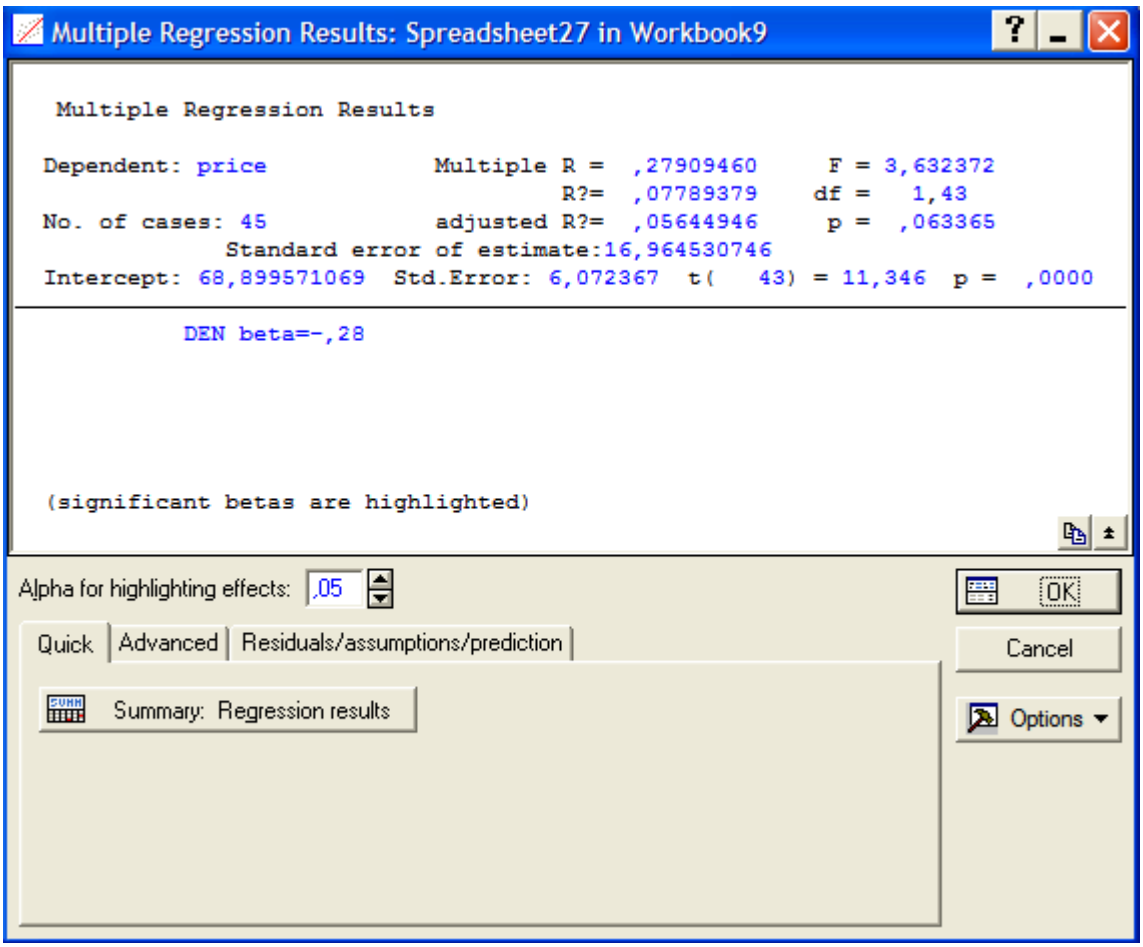


Рисунок 23 – Результат регрессии

В итоге получаем следующую таблицу, изображенную на рисунке 24.

Regression Summary for Dependent Variable: price (Spreadsheet27 in Workbook9)
R= ,27909460 R?= ,07789379 Adjusted R?= ,05644946
F(1,43)=3,6324 p<,06336 Std.Error of estimate: 16,965

N=45	Beta	Std.Err. of Beta	B	Std.Err. of B	t(43)	p-level
Intercept			68,89957	6,072367	11,34641	0,000000
DEN	-0,279095	0,146439	-0,26378	0,138402	-1,90588	0,063365

Рисунок 24 – Таблица анализа коэффициентов

По данным таблицы построим уравнение регрессии. Свободный член равен 68,89, коэффициент регрессии -0,26:

$$y = 68{,}90 - 0{,}26 \cdot x_2 \qquad (5)$$

Теперь найдем множественную корреляцию результативного признака y и всех представленных независимых факторов. Для этого заново выбираем в главном меню Статистика – Множественная регрессия (рисунок 20).

В открывшимся окне переходим на вкладку Расширенный. Ставим галочку «Посмотреть описательную статистику» и инициируем Variables (Переменные).

В новом диалоговом окне (рисунок 25) в левой колонке указываем зависимую переменную (Dependent), а в правой – независимые переменные (Independent). В качестве

зависимой переменной выбираем **цену товара,** в качестве независимых – все имеющиеся факторы, как показано на рисунке. Нажимаем кнопку ОК.

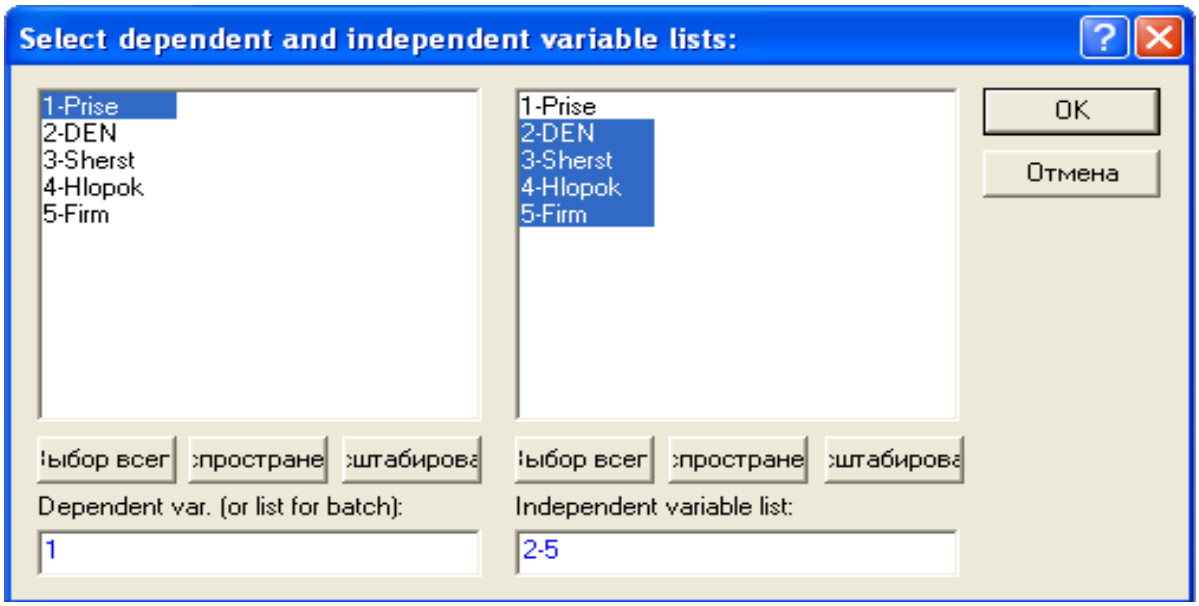


Рисунок 25 – Выбор зависимостей и множества независимых переменных

В новом окне (рисунок 26) переходим на вкладку Расширенный и выбираем Correlations (Корреляции):

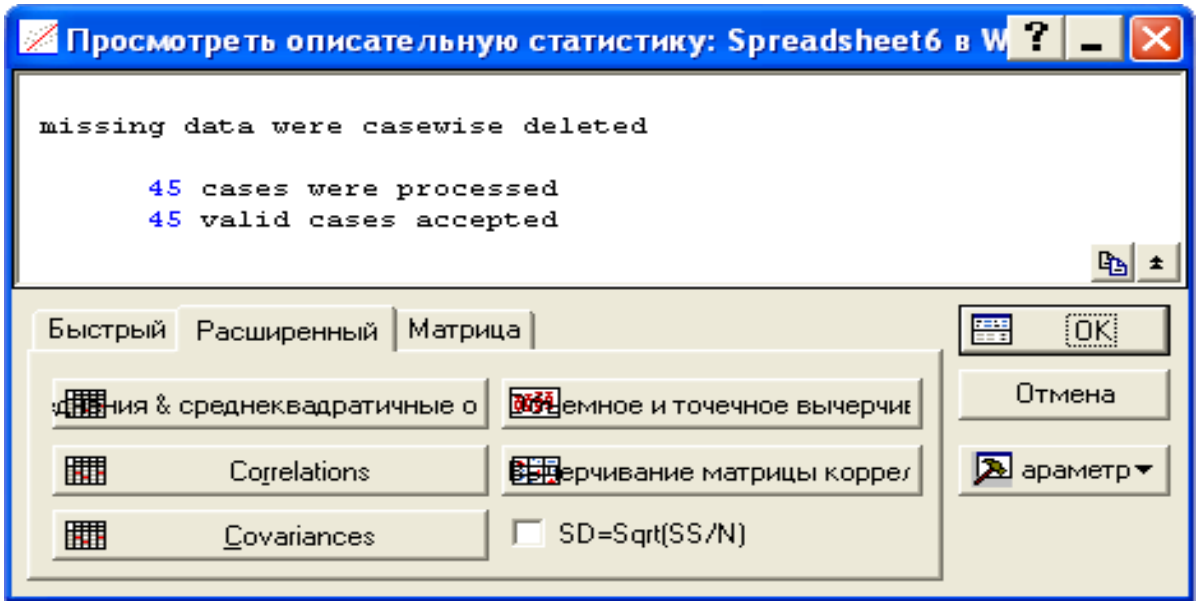


Рисунок 26 – Просмотр описательной статистики

В результате получаем матрицу парных коэффициентов корреляции (рисунок 27). Если в матрице присутствуют мультиколлинеарные факторы, (то есть превышение парным коэффициентом корреляции величины 0,8), то для получения корректного регрессионного уравнения в каждой такой паре необходимо избавиться от того фактора, который наименее влияет на результативный. В нашем случае проявления факторами мультиколлинеарности отсутствует. Однако имеются факторы, практически не влияющие на цену. Их также следует исключить для получения более адекватной модели.

Переменная	Корреляции (Spreadsheet6 в Workbook1) DEN	Sherst	Hlopok	Firm	Prise
DEN	1.000000	-0.421886	0.435579	-0.103535	-0.279095
Sherst	-0.421886	1.000000	-0.667259	0.060901	-0.085989
Hlopok	0.435579	-0.667259	1.000000	-0.439123	0.097683
Firm	-0.103535	0.060901	-0.439123	1.000000	0.060980
Prise	-0.279095	-0.085989	0.097683	0.060980	1.000000

Рисунок 27 – Матрица парных коэффициентов корреляции

В данном случае мы исключим фактор X4 (Фирма-производитель), так как среди рассматриваемых факторных признаков он оказывает на цену наименьшее влияние (коэффициент парной корреляции составляет 0,06).

Для получения уравнения регрессии, описывающего влияние факторов производства на цену товара, проведем в **Statistica** многофакторный регрессионный анализ. Для этого в окне «Посмотреть описательную статистику…» нажмем кнопку Отмена и вернемся в окно «Составная линейная регрессия». *Снимем флажок* с опции «Посмотреть описательную статистику». В Variables, удерживая клавишу Ctrl, выберем те факторы, которые остались после исключения фактора x_4 (рисунок 28).

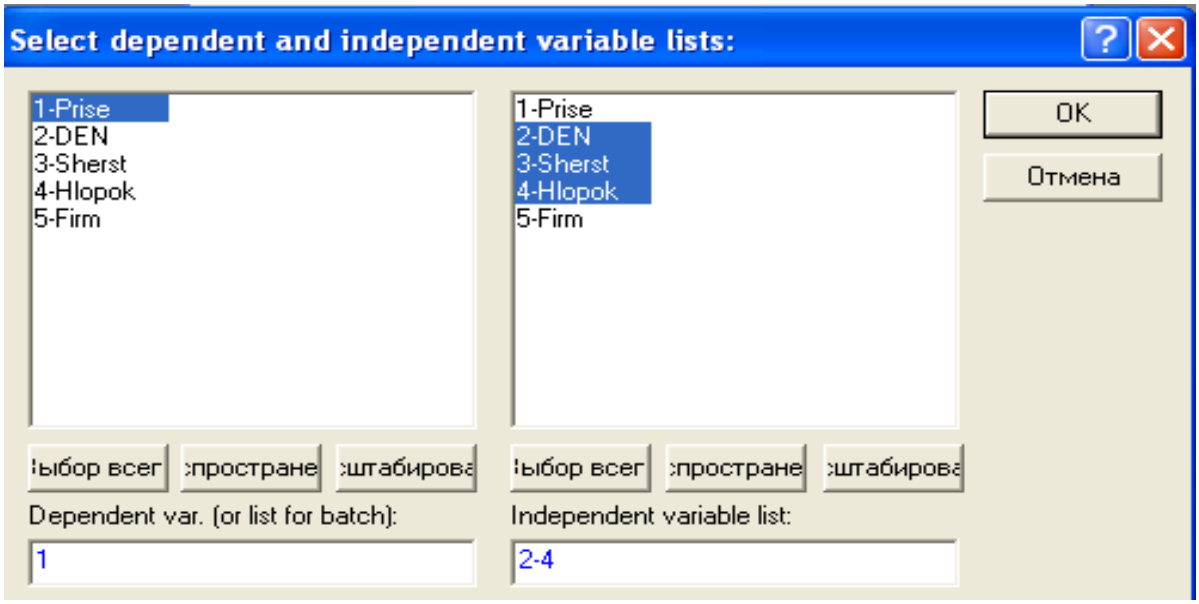


Рисунок 28 – Повторный выбор факторов

Нажав кнопку ОК, перейдем к результатам построения модели (рисунок 29).

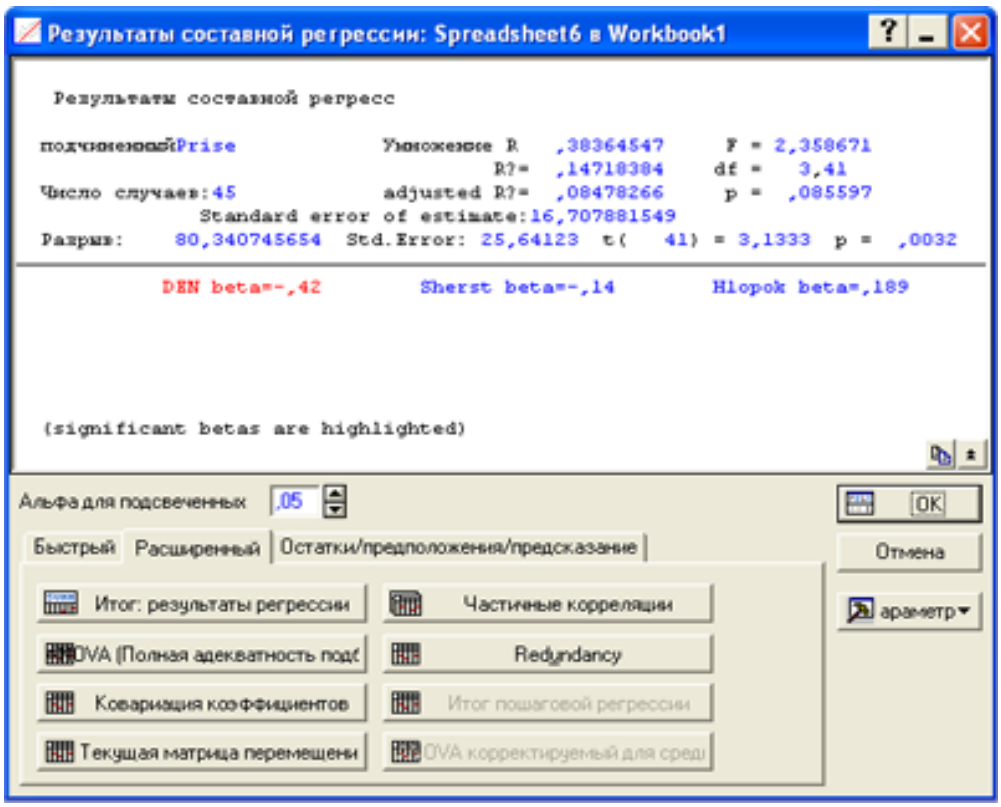


Рисунок 29 – Результаты составной регрессии

После нажатия на кнопку Итог: результаты регрессии, в рабочей области WorkBook получим две таблички: оцененные параметры модели и основные показатели адекватности построения регрессии.

Рассмотрим показатели адекватности построения модели (рисунок 30).

Statistic	Summary St Значени
Умножение R	0.38365
Multiple R?	0.14718
Adjusted R?	0.08478
F(3,41)	2.35867
p	0.03560
Std.Err. of Estimate	16.70788

Рисунок 30 – Показатели адекватности построенной модели

Умножение R (Множественный коэффициент корреляции МКК) является обобщением коэффициента линейной парной корреляции и отражает тесноту связи между зависимой переменной и одновременно всеми учтенными в модели независимыми переменными. Множественный коэффициент корреляции всегда неотрицателен и изменяется от 0 до 1. Чем ближе значение R к 1, тем больше одновременное влияние оказывают независимые переменные. В данном случае МКК равен 0,38365. Он показывает, что связь между вариацией результативного показателя *y* и вариацией факторных признаков средняя.

Multiple R (Множественный коэффициент детерминации) измеряет долю полной вариации переменной *y*, объясняемую множественной регрессией. Величина изменяется от 0 до 1. Согласно полученным результатам, лишь 14% вариаций переменной *y* объясняется задействованными факторами.

Adjusted R (Скорректированный коэффициент детерминации) – неубывающая функция от количества факторов, входящих в модель. Может быть использован для выбора лучшей модели.

P (Вероятность) – если значение P меньше принятого значения α (альфа), то гипотеза о равенстве всех коэффициентов регрессии нулю отвергается. В нашем случае значение вероятности попадает в необходимые рамки.

Рассмотрим вторую таблицу, содержащую параметры модели (рисунок 31).

	Regression Summary for Dependent Variable: Prise (Spreadshee R= ,38364547 R?= ,14718384 Adjusted R?= ,08478266 F(3,41)=2,3587 p<,08560 Std.Error of estimate: 16,708					
N=45	**Бета**	Std.Err. of Beta	B	Std.Err. of B	t(41)	p-level
ОТРЕЗОК			80.34075	25.64123	3.13326	0.003188
DEN	-0.419092	0.163382	-0.39609	0.15442	-2.56510	0.014071
Sherst	-0.136655	0.197454	-0.16476	0.23807	-0.69208	0.492786
Hlopok	0.189047	0.198879	0.41845	0.44021	0.95056	0.347397

Рисунок 31 – Таблица параметров

В четвертом столбце B содержатся значения параметров регрессионного уравнения. Таким образом, мы получили следующее уравнение:

$$y = 80{,}34 - 0{,}396 \cdot x_1 - 0{,}165 \cdot x_2 + 0{,}419 \cdot x_3 . \qquad (6)$$

Полученные значения можно интерпретировать следующим образом. Если при прочих равных условиях расход шерсти на производство носков увеличится на 1%, то цена товара снизится на 0,165 руб. Аналогично и с другими параметрами.

В пятом столбце (Std.Err. of B) указаны стандартные ошибки коэффициентов уравнения. Стандартные ошибки показывают статистическую надежность коэффициента. Значения стандартных ошибок используются для построения доверительных интервалов.

Задача 2

Рассмотрим решение задачи о влиянии метеорологических условий на урожайность некоторой сельскохозяйственной культуры, например яровой пшеницы.

На основе собранных данных о метеорологических условиях местности необходимо выявить влияние погодных условий на урожайность яровой пшеницы в хозяйстве и на основе полученных результатов провести экономический анализ, а также осуществить прогноз на перспективу. На результативный признак (урожайность) оказывают влияние множество факторов. К показателям погодных (метеорологических) условий, оказывающих влияние на урожайность, относятся температура, количество осадков, высота снежного покрова и т.д. Данные для решения задачи представлены в таблице 2.

Необходимо найти коэффициенты корреляции всех факторов. Для выполнения расчетов занесем значения приведенных выше факторов в программный комплекс Statistica 6.0 и проведем множественную регрессию.

Для запуска программного комплекса Statistica 6.0 необходимо кликнуть по ярлыку  на рабочем столе, либо запустить программу из меню «Пуск».

После запуска программы Statistica появится **Рабочее Окно** системы (рисунок 32).

Таблица 2 – Исходные данные к задаче

Номер года	Урожайность, ц/га	Запас влаг в метровом слое почвы, мм		Высота снежного покрова, см		Количество осадков, мм				Среднемесячная температура, C^o	
		Май	Июнь	Март	Январь	Май	Июнь	Июль	Август	Июль	Август
	Y	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
1.	11,5	86	74	50,4	43,3	48	8	24	38	22,5	20,1
2.	11,5	91	140	37	41,3	39	71	57	43	19	16,2
3.	5,7	68	85	41,7	29,3	14	30	23	41	19,9	16,3
4.	23,6	96	73	64	60	27	11	30	44	17,3	15,9
5.	6,7	59	160	49	45	8	86	100	98	19,2	14,6
6.	11,1	93	164	50	39	19	59	85	45	15,5	17,3
7.	7	83	100	33	30	41	47	3	30	21,1	18,4
8.	13	85	83	47	44	31	22	23	50	21,2	16,3
9.	14,4	194	165	51	49	106	43	50	4	18,1	16,4
10.	5,4	191	158	73	57	2	18	30	31	23,3	19,2

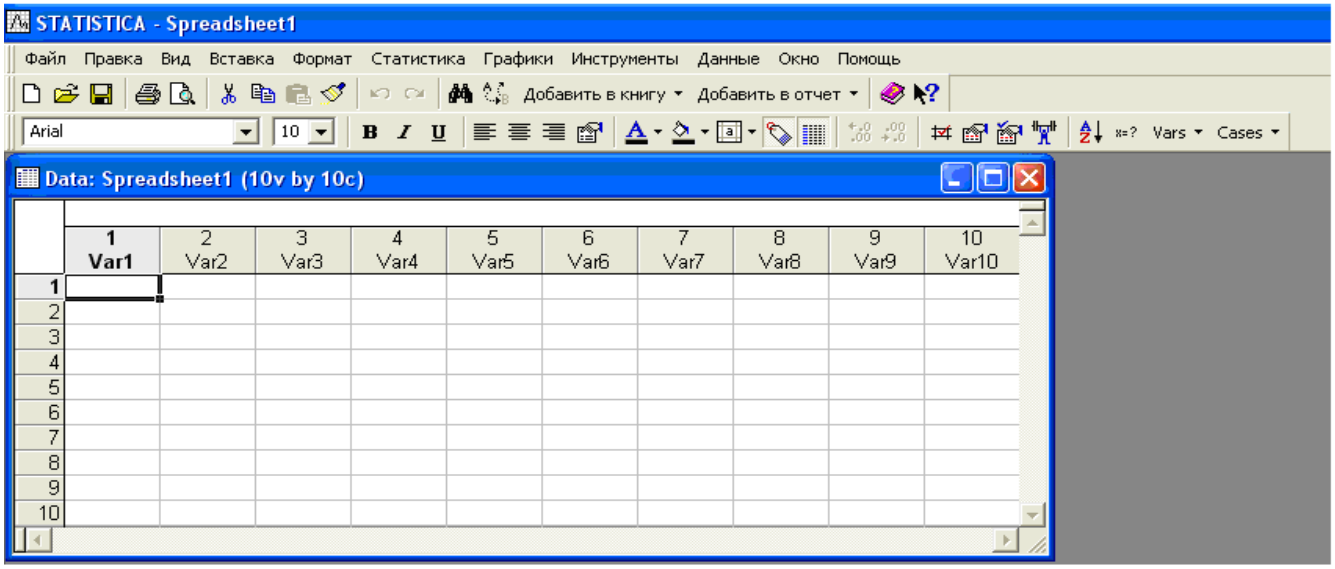


Рисунок 32 – Рабочее окно системы Statistica

Необходимо перенести данные из таблицы 1 (набранной в Excel) в электронную таблицу программы Statistica. Для этого сначала необходимо добавить в электронную таблицу программы Statistica дополнительный столбец, так как в таблице 1 у нас 11 столбцов с данными, а в таблице Statistica лишь 10. Среди команд главного меню программного комплекса Statistica выбираем **Вставка – Добавить переменные**. Откроется диалог **Add Variables** для добавления в электронную таблицу дополнительных столбцов с переменными (рисунок 33).

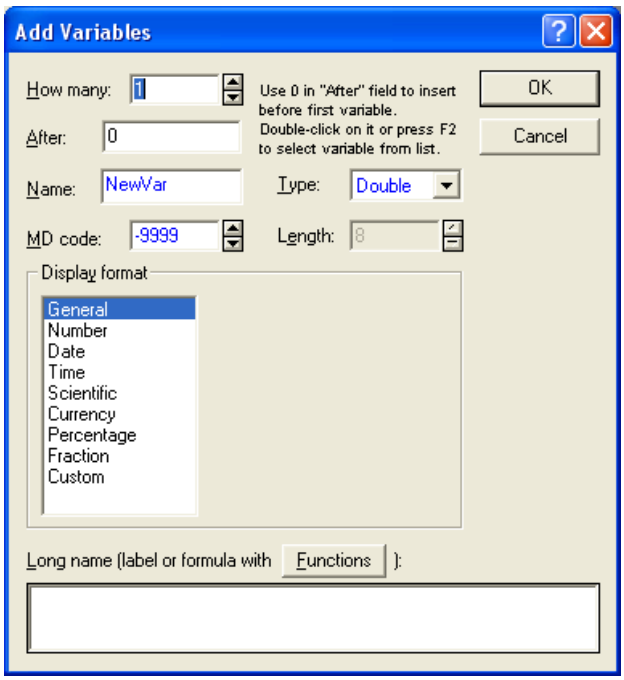


Рисунок 33 – Диалоговое окно для добавления дополнительных столбцов с переменными.

Нажимаем ОК и получаем электронную таблицу уже с 11 столбцами данных (рисунок 34).

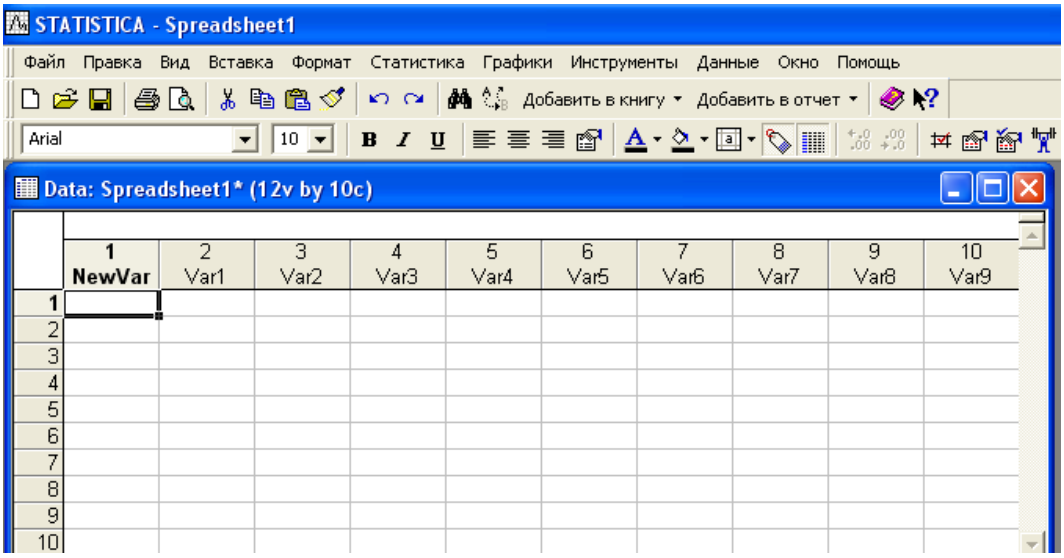


Рисунок 34 – Подготовленная электронная таблица

Копируем исходные данные для задачи из таблицы 2 в электронную таблицу. Кликая двойным щелчком по каждой из ячеек в верхней строке таблицы для каждого столбца вводим название переменных, при этом следует учитывать, что в столбце **New Var** будет вписана *Урожайность*. Например, кликнув на ячейку **Var 1,** открывается диалоговое окно **Variable 1 (Переменная 1)**, где в поле **Name (Имя)** задается название столбца. Проделав указанные действия получаем следующую таблицу (рисунок 35).

	1 Урожайность	2 Май 1	3 Июнь 1	4 Март	5 Январь	6 Май 2	7 Июнь 2	8 Июль 1	9 Август 1	10 Июль 2	11 Август 2
1	11,5	86	74	50,4	43,3	48	8	24	38	22,5	20,1
2	11,5	91	140	37	41,3	39	71	57	43	19	16,2
3	5,7	68	85	41,7	29,3	14	30	23	41	19,9	16,3
4	23,6	96	73	64	60	27	11	30	44	17,3	15,9
5	6,7	59	160	49	45	8	86	100	98	19,2	14,6
6	11,1	93	164	50	39	19	59	85	45	15,5	17,3
7	7	83	100	33	30	41	47	3	30	21,1	18,4
8	13	85	83	47	44	31	22	23	50	21,2	16,3
9	14,4	194	165	51	49	106	43	50	4	18,1	16,4
10	5,4	191	158	73	57	2	18	30	31	23,3	19,2

Рисунок 35 – Заполненная электронная таблица

Далее в меню программы Statistica выбираем пункт **Статистика -Множественная регрессия**. Появляется диалоговое окно, в котором необходимо кликнуть по кнопке **Variables,** в открывшемся окне в левом столбце указываем зависимую переменную (Dependent var.), а в правом независимые (Independent variable list) (рисунок 36).

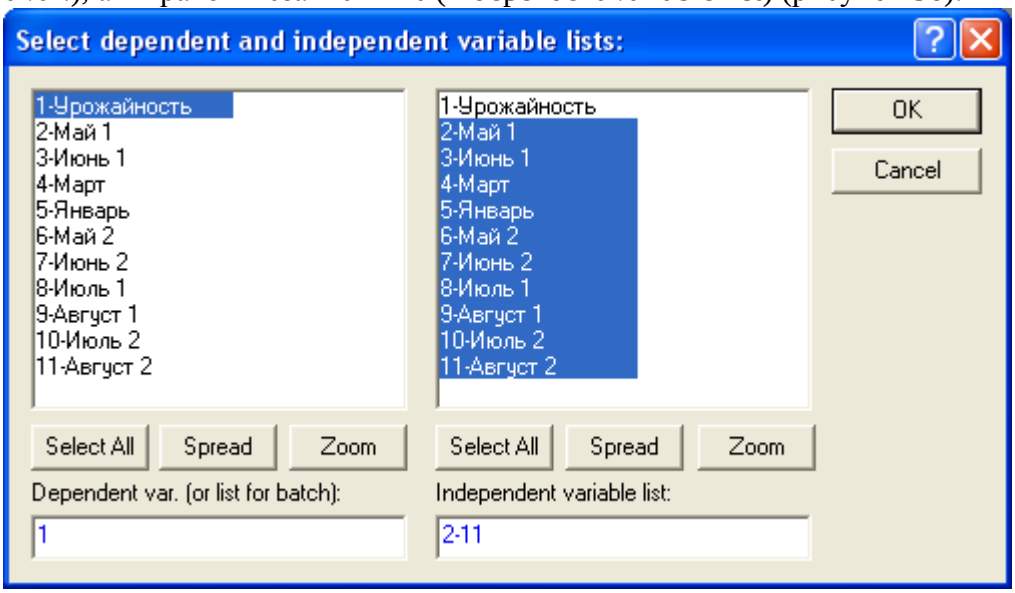


Рисунок 36 – Выделенные зависимая и независимые переменные

Нажимаем ОК, в открытом диалоговом на вкладке **Advanced (Настройки)** ставим галочку напротив пункта **Review descriptive statistics, correlation matrix**. Нажимаем ОК, в следующем окне на вкладке **Quick** кликаем по кнопке **Correlations**. В результате получаем матрицу парных коэффициентов корреляции, записанную в таблице 3.

Таблица 3 – Матрица парных коэффициентов корреляции

	Май1	Июнь1	Март	Январь	Май2	Июнь2	Июль1	Авг1	Июль2	Авг2	Урож
Май1	1,00	0,48	0,57	0,55	0,43	-0,26	-0,10	-0,67	0,15	0,30	0,06
Июнь1	0,48	1,00	0,16	0,15	0,07	0,66	0,72	0,04	-0,30	-0,18	-0,32
Март	0,57	0,16	1,00	**0,87**	-0,24	-0,48	0,06	-0,06	0,11	0,18	0,27
Январь	0,55	0,15	**0,87**	1,00	0,02	-0,34	0,12	-0,04	-0,01	-0,03	0,56
Май2	0,43	0,07	-0,24	0,02	1,00	-0,05	-0,14	-0,65	-0,16	0,04	0,36
Июнь2	-0,26	0,66	-0,48	-0,34	-0,05	1,00	0,75	0,48	-0,43	-0,54	-0,34
Июль1	-0,10	0,72	0,06	0,12	-0,14	0,75	1,00	0,56	-0,58	-0,52	-0,06
Авг1	-0,67	0,04	-0,06	-0,04	-0,65	0,48	0,56	1,00	-0,08	-0,49	-0,19
Июль2	0,15	-0,30	0,11	-0,01	-0,16	-0,43	-0,58	-0,08	1,00	0,59	-0,48
Авг2	0,30	-0,18	0,18	-0,03	0,04	-0,54	-0,52	-0,49	0,59	1,00	-0,24
Урож	0,06	-0,32	0,27	0,56	0,36	-0,34	-0,06	-0,19	-0,48	-0,24	1,00

Одним из основных препятствий эффективного применения множественного регрессионного анализа является коллинеарность или мультиколлинеарность, которая возникает в случаях существования достаточно тесных линейных статистических связей между объясняющими переменными. Если две переменные находятся между собой в линейной зависимости, то говорят, что они коллинеарны.

О наличии мультиколлинеарности факторов говорят, когда более чем два фактора связаны между собой линейной зависимостью, т.е. имеет место совокупное воздействие факторов друг на друга. Поскольку одним из условий построения множественной регрессии является независимость действия факторов, то если факторы явно коллинеарны, то один из них рекомендуется исключить из регрессии. Предпочтение при этом отдается не фактору, более тесно связанному с результатом, а тому фактору, который при достаточно тесной связи с результатом имеет наименьшую тесноту связи с другими факторами.

Точных количественных критериев для определения наличия или отсутствия реальной мультиколлинеарности не существует. Тем не менее существуют некоторые эвристически рекомендации по ее выявлению. В первую очередь анализируют матрицу парных коэффициентов корреляции (ту ее часть, которая относится к объясняющим переменным).

В этом отношении существует ряд мнений и о наличии коллинеарности (мультиколлинеарности) говорят в случае, если: коэффициент парной корреляции $r_{x_i x_j}$ принимает значение:

а) $r_{x_i x_j} \geq 0{,}7$[1)];

б) $\left|r_{x_i x_j}\right| \geq 0{,}75 - 0{,}80$;

в) $\left|r_{x_i x_j}\right| \geq 0{,}80$.

В таблице 3 жирным шрифтом выделены значения парной корреляции для мультиколлинеарных факторов (согласно третьему подходу, превышение парным коэффициентом корреляции величины 0,8).

В данном случае мы исключим фактор «май1» (среднемесячный запас влаги в почве в мае), так как среди рассматриваемых факторных признаков он оказывает на урожайность наименьшее влияние (коэффициент парной корреляции составляет 0,06). Кроме того, исключаем из последующего рассмотрения и фактор «март» (высота снежного покрова в марте), так как этот фактор мультиколлинеарен с фактором «январь» и оказывает на результативный показатель меньшее воздействие (коэффициент парной корреляции составляет 0,27).

Для этого нажимаем **в рабочей книге (Workbook 1)** на кнопку **Review descriptive statistics** (рисунок 37).

Variable										
Май 1	1,000000	0,483769	0,570912	0,545482	0,426686	-0,261372	-0,101023	-0,671751	0,146376	0,297052
Июнь 1	0,483769	1,000000	0,158913	0,154262	0,072717	0,656604	0,717254	0,038958	-0,298458	-0,181244
Март	0,570912	0,158913	1,000000	0,866564	-0,235814	-0,481835	0,056508	-0,057589	0,111148	0,184477
Январь	0,545482	0,154262	0,866564	1,000000	0,019878	-0,336605	0,121914	-0,036313	-0,007846	-0,033638
Май 2	0,426686	0,072717	-0,235814	0,019878	1,000000	-0,050023	-0,138375	-0,648862	-0,157916	0,039565
Июнь 2	-0,261372	0,656604	-0,481835	-0,336605	-0,050023	1,000000	0,749561	0,475646	-0,434885	-0,544530
Июль 1	-0,101023	0,717254	0,056508	0,121914	-0,138375	0,749561	1,000000	0,560622	-0,584998	-0,521962
Август 1	-0,671751	0,038958	-0,057589	-0,036313	-0,648862	0,475646	0,560622	1,000000	-0,081380	-0,486425
Июль 2	0,146376	-0,298458	0,111148	-0,007846	-0,157916	-0,434885	-0,584998	-0,081380	1,000000	0,591990
Август 2	0,297052	-0,181244	0,184477	-0,033638	0,039565	-0,544530	-0,521962	-0,486425	0,591990	1,000000
Урожайность	0,056179	-0,322267	0,266107	0,556875	0,363554	-0,338616	-0,056491	-0,186671	-0,480937	-0,242092

Рисунок 37 – Рабочая книга с данными

В открывшемся окне нажимаем **Cancel (Отмена),** затем кликаем по кнопке **Variables** и в следующем окне **Select Dependent and independent variable list** в правом столбце исключаем из выделенных переменных, нажав клавишу Ctrl, факторы «май 1» и «март» (рисунок 38).

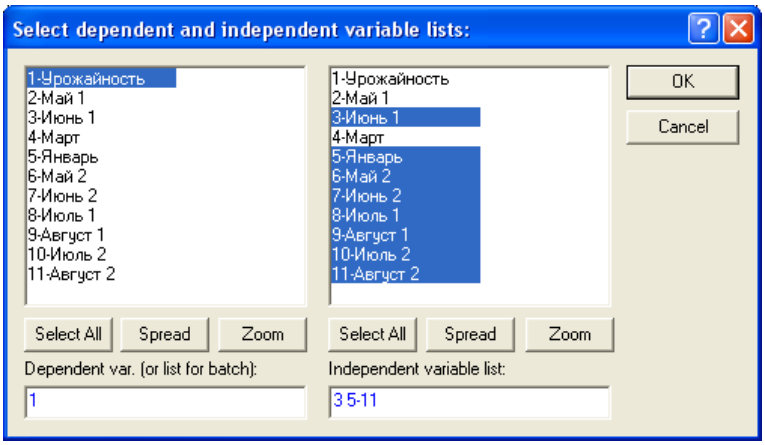


Рисунок 38 – Таблица с исключенными факторами «май 1» и «март»

Нажимаем ОК, снимаем галочку с пункта **Review descriptive statistics**, нажимаем ОК, а затем **Summary: regression results.**

В результате получаем две таблички: оцененные параметры модели и основные показатели адекватности построения регрессии.

Чтобы просмотреть показатели адекватности построения модели необходимо в рабочей книге перейти на вкладку **Summary statistics** (рисунок 39) и записать данные в отдельную таблицу 4.

Summary Statistics; DV: Урожайность (Spreadsheet1.sta)

Statistic	Value
Multiple R	0,990586
Multiple R?	0,981261
Adjusted R?	0,831347
F(8,1)	6,545517
p	0,293901
Std.Err. of Estimate	2,244217

Рисунок 39 – Показатели адекватности построения модели

Таблица 4 – Показатели адекватности множественного уравнения регрессии

	Значение
Multiple R	0,990586
Multiple R?	0,981261
Adjusted R?	0,831347
F (8,1)	6,545517
P	0,293901
Std.Err.of Estimate	2,244217

Multiple R (Множественный коэффициент корреляции) является обобщением коэффициента линейной парной корреляции и отражает тесноту связи между зависимой переменной и одновременно всеми учтенными в модели независимыми переменными. Множественный коэффициент корреляции всегда неотрицателен и изменяется от 0 до 1. Чем

ближе значение R к 1, тем более одновременное влияние оказывают независимые переменные.

В данном случае множественный коэффициент корреляции равен 0,990586. Он показывает, что связь между вариацией результативного показателя «урожайность» и вариацией факторных признаков сильная.

Multiple R? (Множественный коэффициент детерминации) измеряет долю полной вариации переменной *y*, объясняемую множественной регрессией. Величина изменяется от 0 до 1.

Согласно полученным результатам (таблица 4.4) R^2=0,981261, это свидетельствует о том, что 98,13% вариации переменной «урожайность» объясняется задействованными факторами.

Adjusted R? (Скорректированный коэффициент детерминации) – неубывающая функция от количества факторов, входящих в модель. Может быть использован для выбора лучшей модели, Adjusted R =0,831347.

F(8,l) – F-статистика Фишера служит для проверки модели на адекватность. Необходимо сопоставить табличное значение F-критерия при $\alpha = 0,05$ и $\nu_1 = 8$, $\nu_2 = 1$ с фактическим значением F(8, 1) = 6,545517. Табличное значение F-критерия можно посмотреть в статистических таблицах Стьюдента-Фишера на пересечении 8-го столбца и 1-ой строки. При данных условиях табличное значение критерия составляет 238,9, т.е. получаем Fтабл. > Fфакт., следовательно модель статистически незначима.

P (Вероятность) – если значение вероятности меньше принятого значения α (в нашем случае $\alpha = 0,05$), то нулевая гипотеза о равенстве всех коэффициентов регрессии нулю отвергается. Из таблицы 4.4 видно, что P > 0,05, следовательно гипотеза о равенстве всех коэффициентов регрессии нулю не отвергается.

Чтобы рассмотреть вторую таблицу, содержащую параметры модели, для этого необходимо перейти на вкладку **Regression Summary for Dependent Variable** (рисунок 40).

N=10	**Beta**	Std.Err. of Beta	B	Std.Err. of B	t(1)	p-level	
Intercept			23,19355	14,46361	1,60358	0,354976	
Июнь 1	-0,905164	0,650849	-0,12230	0,08794	-1,39074	0,396863	
Январь	0,821187	0,245279	0,44990	0,13438	3,34797	0,184781	
Май 2	0,220782	0,314926	0,04089	0,05833	0,70106	0,610747	
Июнь 2	0,413965	0,511974	0,08573	0,10603	0,80857	0,567135	
Июль 1	0,024747	0,727581	0,00444	0,13050	0,03401	0,978356	
Август 1	-0,180180	0,737966	-0,04203	0,17216	-0,24416	0,847547	
Июль 2	-0,600253	0,380998	-1,36796	0,86828	-1,57547	0,360050	
Август 2	0,118772	0,249186	0,38635	0,81057	0,47664	0,716840	

Рисунок 40 – Результаты оценивания множественного уравнения регрессии

Переносим данные в отдельную таблицу 5 и проведем необходимый анализ.

Таблица 5 – Результаты оценивания множественного уравнения регрессии

	Бета	Std.Err.of Beta	B	Std.Err.of B	t(1)	p-level
ОТРЕЗОК			23,19355	14,46361	1,60358	0,354976
июнь1	-0,905164	0,650849	-0,12230	0,08794	-1,39074	0,396863
январь	0,821187	0,245279	0,44990	0,13438	3,34797	0,184781
май2	0,220782	0,314926	0,04089	0,05833	0,70106	0,610747
июнь2	0,413965	0,511974	0,08573	0,10603	0,80857	0,567135
июль1	0,024747	0,727581	0,00444	0,13050	0,03401	0,978356
авг1	-0,180180	0,737966	-0,04203	0,17216	-0,24416	0,847547
июль2	-0,600253	0,380998	-1,36796	0,86828	-1,57547	0,360050
авг2	0,118772	0,249186	0,38635	0,81057	0,47664	0,716840

Рассмотрим результаты оценки параметров уравнения регрессии по столбцам. В первом столбце перечислены члены регрессионного уравнения, ОТРЕЗОК – свободный член уравнения.

Во втором столбце (Бета) содержатся β-коэффициенты. Они являются абстрактными величинами, указывающими на сколько среднеквадратических отклонений увеличится зависимая переменная при изменении соответствующей независимой переменной на 1 среднеквадратическое отклонение. На практике данный показатель используется для выявления фактора, оказывающего наибольшее влияние на зависимую переменную. В нашем случае наибольшее (положительное) влияние оказывает показатель «январь» – высота снежного покрова в январе (β = 0,821187).

В четвертом столбце (B) содержатся значения параметров регрессионного уравнения. Таким образом, мы получили следующее уравнение:

$$y = 23{,}194 - 0{,}122x_2 + 0{,}450x_4 + 0{,}041x_5 + 0{,}086x_6 + 0{,}004x_7 - \\ - 0{,}042x_8 - 1{,}368x_9 + 0{,}386x_{10}. \quad (7)$$

Полученные значения можно интерпретировать таким образом: Если при прочих равных условиях (a_2 = -0,122) запас влаги в метровом слое почвы в июне увеличится на 1 мм, то урожайность уменьшится на 0,122 ц/га. Для четвертого параметра: если при прочих равных условиях (a_4 = 0,450) высота снежного покрова в январе увеличится на 1 мм, то урожайность увеличиться на 0,450 ц/га. Аналогично, можно сделать выводы по всем остальным параметрам полученного уравнения регрессии.

В пятом столбце (Std.Err. of B) указаны стандартные ошибки коэффициентов уравнения. Стандартные ошибки показывают статистическую надежность коэффициента. Значения стандартных ошибок используются для построения доверительных интервалов.

t(1) – выводит расчетное значение t-статистики Стьюдента. Ее значение используется для проверки значимости соответствующего коэффициента путем сравнения с ним табличного значения t-критерия при a = 0,05 и df = l. Данное значение равно 12,706, сравним его с расчетными значениями:

a_0 = |1,60358| < 12,706 – параметр статистически незначим;
a_2 = |-1,39074| < 12,706 – параметр статистически незначим;
a_4 = |3,34797| < 12,706 – параметр статистически незначим;
a_5 = |0,70106| < 12,706 – параметр статистически незначим;
a_6 = |0,80857| < 12,706 – параметр статистически незначим;
a_7 = |0,03401| < 12,706 – параметр статистически незначим;
a_8 = |-0,24416| < 12,706 – параметр статистически незначим;
a_9 = |-1,57547| < 12,706 – параметр статистически незначим;
a_{10} = |0,47664| < 12,706 – параметр статистически незначим.

Полученная множественная регрессионная модель незначима по всем параметрам. В этом случае, необходимо исключить из рассмотрения фактор «июль1» (количество осадков в июле), так как среди рассматриваемых факторных признаков он оказывает на урожайность наименьшее влияние (t = 0,03401) и повторить описанные операции сначала (следует отметить, что данный фактор можно было исключить еще на первом этапе (стр.84), так как коэффициент парной корреляции между «июль1» и «июль2» равен 0,75 и между «июль1» и «июль2» равен 0,72, а ряд исследователей считает, что факторы будут коллинеарны, если коэффициент парной корреляции превышает значение 0,7).

В результате получим значения, записанные в таблице 6.

В данном случае множественный коэффициент корреляции равен 0,99058, т.е. связь между вариацией результативного показателя «урожайность» и вариацией факторных признаков сильная.

Multiple R? = 0,98124, это свидетельствует о том, что 98,12% вариации переменной «урожайность» объясняется задействованными факторами.

Adjusted R? = 0,91558.

Таблица 6 – Показатели адекватности множественного уравнения регрессии

	Значение
Умножение R	0,99058
Multiple R?	0,98124
Adjusted R?	0,91558
F (7,2)	14,94357
P	0,06414
Std.Err.of Estimate	1,58782

F(7,2) – при данных условиях табличное значение критерия составляет 19,35, т.е. получаем Fтабл. > Fфакт., следовательно модель статистически незначима.

Рассмотрим таблицу 7, содержащую параметры модели.

Таблица 7 – Результаты оценивания множественного уравнения регрессии

	Бета	Std.Err.of Beta	B	Std.Err.of B	t(2)	p-level
ОТРЕЗОК			23,23202	10,20188	2,27723	0,150487
июнь1	-0,885918	0,227523	-0,11970	0,03074	-3,89376	0,060075
январь	**0,819181**	**0,168444**	**0,44880**	**0,09228**	**4,86323**	**0,039776**
май2	0,228317	0,158372	0,04229	0,02933	1,44165	0,286132
июнь2	0,405791	0,319842	0,08404	0,06624	1,26872	0,332219
авг1	-0,157604	0,228197	-0,03677	0,05324	-0,69065	0,561171
июль2	**-0,611559**	**0,131717**	**-1,39372**	**0,30018**	**-4,64298**	**0,043391**
авг2	0,122202	0,161226	0,39750	0,52444	0,75795	0,527614

Рассмотрим результаты оценки параметров уравнения регрессии по столбцам. Во втором столбце (Бета) содержатся β-коэффициенты, в нашем случае наибольшее (положительное) влияние оказывает показатель «январь» – высота снежного покрова в январе (β = 0,819181).

В четвертом столбце (B) содержатся значения параметров регрессионного уравнения. Таким образом, мы получили следующее уравнение:

$$y = 23{,}232 - 0{,}120x_2 + 0{,}449x_4 + 0{,}042x_5 + 0{,}084x_6 - 0{,}037x_8 - \\ - 1{,}394x_9 + 0{,}398x_{10}. \quad (8)$$

t(2) – расчетное значение t-статистики Стьюдента равно 4,3020; сравним его с расчетными значениями:

a_0 = |2,27723| < 4,3020- параметр статистически незначим;

a_2 = |-3,89376 |< 4,3020- параметр статистически незначим;

a_4 = |**4,86323**| > 4,3020- параметр статистически значим;

a_5 = |1,26872| < 4,3020- параметр статистически незначим;

a_6 = |1,44165| <4,3020- параметр статистически незначим;

a_8 = |-0,69065| < 4,3020- параметр статистически незначим;

a_9 = |**-4,64298**| > 4,3020- параметр статистически значим;

a_{10} = |0,75795| < 4,3020- параметр статистически незначим.

Полученная множественная регрессионная модель имеет параметры, которые статистически незначимы. В этом случае, необходимо исключить из рассмотрения фактор «август2» (среднемесячная температура в августе), так как среди рассматриваемых факторных признаков он оказывает на урожайность наименьшее влияние (t = 0,75795) и повторить описанные операции сначала.

В результате получим значения, записанные в таблице 8.

Таблица 8 – Показатели адекватности множественного уравнения регрессии

	Значение
Умножение R	0,98785
Multiple R?	0,97585
Adjusted R?	0,92755
F (6,3)	20,20406
P	0,01595
Std.Err.of Estimate	1,47091

В данном случае множественный коэффициент корреляции равен 0,98785, т.е. связь между вариацией результативного показателя «урожайность» и вариацией факторных признаков сильная.

Multiple R? = 0,97585, это свидетельствует о том, что 97,59% вариации переменной «урожайность» объясняется задействованными факторами.

Adjusted R? =0,92755.

F(6,3) – при данных условиях табличное значение критерия составляет 8,94, т.е. получаем Fтабл. < Fфакт., следовательно модель статистически значима.

Рассмотрим таблицу 9, содержащую параметры модели (8).

Таблица 9 – Результаты оценивания множественного уравнения регрессии

	Бета	Std.Err.of Beta	B	Std.Err.of B	t(3)	p-level
ОТРЕЗОК			**28,97959**	**6,322205**	**4,58378**	**0,019497**
июнь1	**-0,832092**	**0,200241**	**-0,11242**	**0,027054**	**-4,15546**	**0,025337**
январь	**0,779062**	**0,148138**	**0,42682**	**0,081160**	**5,25904**	**0,013394**
май2	0,214454	0,145729	0,03972	0,026992	1,47159	0,237515
июнь2	0,324725	0,279236	0,06725	0,057829	1,16291	0,328962
авг1	-0,187110	0,208297	-0,04365	0,048594	-0,89829	0,435232
июль2	**-0,563312**	**0,106824**	**-1,28377**	**0,243449**	**-5,27326**	**0,013295**

Рассмотрим результаты оценки параметров уравнения регрессии по столбцам. Во втором столбце (Бета) содержатся β-коэффициенты, в нашем случае наибольшее (положительное) влияние оказывает показатель «январь» – высота снежного покрова в январе (β = 0,779062).

В четвертом столбце (B) содержатся значения параметров регрессионного уравнения. Таким образом, мы получили следующее уравнение:

$$y = 28{,}980 - 0{,}112x_2 + 0{,}427x_4 + 0{,}040x_5 + 0{,}067x_6 - 0{,}044x_8 - 1{,}284x_9. \quad (9)$$

t(3) – расчетное значение t-статистики Стьюдента равно 3,1820; сравним его с расчетными значениями:

a_0 = |**4,58378**| > 3,1820- параметр статистически значим;

a_2 = |**-4,15546**| > 3,1820- параметр статистически значим;

a_4 = |**5,25904**| > 3,1820- параметр статистически значим;

a_5 = |1,47159| < 3,1820- параметр статистически незначим;

a_6 = |1,16291| < 3,1820- параметр статистически незначим;

a_8 = |-0,89829| < 3,1820- параметр статистически незначим;

a_9 = |**-5,27326**| > 3,1820- параметр статистически значим.

Полученная множественная регрессионная модель имеет параметры, которые статистически незначимы. В этом случае, необходимо исключить из рассмотрения фактор «июнь2» (количество осадков в июне), так как среди рассматриваемых факторных признаков

он оказывает на урожайность наименьшее влияние (t = 1,16291) и повторить описанные операции сначала.

В результате получим следующие показатели (таблица 10).

Таблица 10 – Показатели адекватности множественного уравнения регрессии

	Значение
Умножение R	0,98233
Multiple R?	0,96496
Adjusted R?	0,92117
F (5,4)	22,03350
P	0,00518
Std.Err.of Estimate	1,53433

В данном случае множественный коэффициент корреляции равен 0,98233, т.е. связь между вариацией результативного показателя «урожайность» и вариацией факторных признаков сильная.

Multiple R? = 0,96496, это свидетельствует о том, что 96,50% вариации переменной «урожайность» объясняется задействованными факторами.

Adjusted R? =0,92117.

F(5,4) – при данных условиях табличное значение критерия составляет 6,26, т.е. получаем Fтабл. < Fфакт., следовательно модель статистически значима.

Результаты оценивания представлены в таблице 11.

Таблица 11 – Результаты оценивания множественного уравнения регрессии

	Бета	Std.Err.of Beta	B	Std.Err.of B	t(4)	p-level
ОТРЕЗОК			**31,34691**	**6,243567**	**5,02067**	**0,007382**
июнь1	**-0,627341**	**0,099487**	**-0,08476**	**0,013442**	**-6,30577**	**0,003234**
январь	**0,643099**	**0,094891**	**0,35233**	**0,051987**	**6,77727**	**0,002474**
май2	0,305482	0,128222	0,05658	0,023749	2,38244	0,075788
авг1	0,009360	0,127091	0,00218	0,029649	0,07364	0,944829
июль2	**-0,614124**	**0,101682**	**-1,39957**	**0,231731**	**-6,03964**	**0,003790**

Рассмотрим результаты оценки параметров уравнения регрессии по столбцам. Во втором столбце (Бета) содержатся β-коэффициенты, в нашем случае наибольшее (положительное) влияние оказывает показатель «январь» - высота снежного покрова в январе (β = 0,643099).

В четвертом столбце (B) содержатся значения параметров регрессионного уравнения. Таким образом, мы получили следующее уравнение:

$$y = 31{,}347 - 0{,}085x_2 + 0{,}352x_4 + 0{,}057x_5 + 0{,}002x_8 - 1{,}400x_9. \quad (10)$$

t(4) – расчетное значение t-статистики Стьюдента равно 2,7760; сравним его с расчетными значениями:

a_0 = |**5,02067**| > 2,7760 – параметр статистически значим;

a_2 = |**-6,30577**| > 2,7760 – параметр статистически значим;

a_4 = |**6,77727**| > 2,7760 – параметр статистически значим;

a_5 = |2,38244| < 2,7760 – параметр статистически незначим;

a_8 = |0,07364| < 2,7760 – параметр статистически незначим;

a_9 = |**-6,03964**| > 2,7760 – параметр статистически значим.

Полученная множественная регрессионная модель имеет параметры, которые статистически незначимы. В этом случае, необходимо исключить из рассмотрения фактор

«август1» (количество осадков в августе), так как среди рассматриваемых факторных признаков он оказывает на урожайность наименьшее влияние (t = 0,07364) и повторить описанные операции сначала.

В результате получим следующие показатели адекватности (таблица 12).

Таблица 12 – Показатели адекватности множественного уравнения регрессии

	Значение
Умножение R	0,98230
Multiple R?	0,96492
Adjusted R?	0,93685
F (4,5)	34,37904
P	0,00079
Std.Err.of Estimate	1,37328

В данном случае множественный коэффициент корреляции равен 0,98230 т.е. связь между вариацией результативного показателя «урожайность» и вариацией факторных признаков по прежнему сильная.

Multiple R? = 0,96492, это свидетельствует о том, что 96,49% вариации переменной «урожайность» объясняется задействованными факторами.

Adjusted R? =0,93685.

F(4,5) – при данных условиях табличное значение критерия составляет 5,19, т.е. получаем Fтабл. < Fфакт., следовательно модель статистически значима.

Результаты оценивания приведены в таблице 13.

Таблица 13 – Результаты оценивания множественного уравнения регрессии

	Бета	Std.Err.of Beta	B	Std.Err.of B	t(5)	p-level
ОТРЕЗОК			**31,55399**	**4,989398**	**6,32421**	**0,001457**
июнь1	**-0,626964**	**0,088926**	**-0,08471**	**0,012015**	**-7,05038**	**0,000887**
январь	**0,642814**	**0,084860**	**0,35218**	**0,046492**	**7,57502**	**0,000636**
май2	**0,299126**	**0,084871**	**0,05540**	**0,015720**	**3,52448**	**0,016839**
июль2	**-0,615779**	**0,088758**	**-1,40334**	**0,202276**	**-6,93774**	**0,000955**

Рассмотрим результаты оценки параметров уравнения регрессии по столбцам. Во втором столбце (Бета) содержатся β-коэффициенты, в нашем случае наибольшее (положительное) влияние оказывает показатель «январь» - высота снежного покрова в январе (β = 0,642814).

В четвертом столбце (B) содержатся значения параметров регрессионного уравнения. Таким образом, мы получили следующее уравнение:

$$y = 31{,}554 - 0{,}085x_2 + 0{,}352x_4 + 0{,}055x_5 - 1{,}403x_9. \quad (11)$$

где x_2 – июнь1; x_4 – январь1; x_5 – май2; x_9 – июль2.

t(5) – расчетное значение t-статистики Стьюдента равно 2,5700; сравним его с расчетными значениями:

a_0 = |**6,32421**| > 2,5700 – параметр статистически значим;

a_2 = |**-7,05038**| > 2,5700 – параметр статистически значим;

a_4 = |**7,57502**| > 2,5700 – параметр статистически значим;

a_5 = |**3,52448**| > 2,5700 – параметр статистически значим;

a_9 = |**-6,93774**| > 2,5700 – параметр статистически значим.

Полученная скорректированная множественная регрессионная модель значима по всем параметрам. При этом показатели Умножение R, Multiple R? и Adjusted R? в скорректированной модели снизились, а Std.Err. of Estimate возросло.

Последний столбец таблицы (p-level) - показывает вероятность принять или отвергнуть гипотезу о равенстве нулю соответствующего коэффициента. Если значение вероятности ниже уровня значимости $\alpha = 0{,}05$, то гипотеза H_0 отвергается и соответствующий коэффициент не равен нулю. В нашем случае, все параметры статистически значимы.

Для получения расчетной урожайности подставим в полученное уравнение множественной регрессии 4.11 фактические параметры погоды за исследуемый период:

$y_1 = 31{,}554 - 0{,}085 \cdot 74 + 0{,}352 \cdot 43{,}3 + 0{,}055 \cdot 48 - 1{,}403 \cdot 22{,}5 = 19{,}6781;$

$y_2 = 31{,}554 - 0{,}085 \cdot 140 + 0{,}352 \cdot 41{,}3 + 0{,}055 \cdot 39 - 1{,}403 \cdot 19 = 16{,}5196;$

$y_3 = 31{,}554 - 0{,}085 \cdot 85 + 0{,}352 \cdot 29{,}3 + 0{,}055 \cdot 14 - 1{,}403 \cdot 19{,}9 = 14{,}6569;$

$y_4 = 31{,}554 - 0{,}085 \cdot 73 + 0{,}352 \cdot 60 + 0{,}055 \cdot 27 - 1{,}403 \cdot 17{,}3 = 29{,}9101;$

$y_5 = 31{,}554 - 0{,}085 \cdot 160 + 0{,}352 \cdot 45 + 0{,}055 \cdot 8 - 1{,}403 \cdot 19{,}2 = 14{,}2084;$

$y_6 = 31{,}554 - 0{,}085 \cdot 164 + 0{,}352 \cdot 39 + 0{,}055 \cdot 19 - 1{,}403 \cdot 15{,}5 = 16{,}2205;$

$y_7 = 31{,}554 - 0{,}085 \cdot 100 + 0{,}352 \cdot 30 + 0{,}055 \cdot 41 - 1{,}403 \cdot 21{,}1 = 13{,}8617;$

$y_8 = 31{,}554 - 0{,}085 \cdot 83 + 0{,}352 \cdot 44 + 0{,}055 \cdot 31 - 1{,}403 \cdot 21{,}2 = 19{,}5804;$

$y_9 = 31{,}554 - 0{,}085 \cdot 165 + 0{,}352 \cdot 49 + 0{,}055 \cdot 106 - 1{,}403 \cdot 18{,}1 = 21{,}7287;$

$y_{10} = 31{,}554 - 0{,}085 \cdot 158 + 0{,}352 \cdot 57 + 0{,}055 \cdot 2 - 1{,}403 \cdot 23{,}3 = 13{,}9961.$

Затем перенесем расчетные и фактические значения в программу Microsoft Excel (рисунок 41).

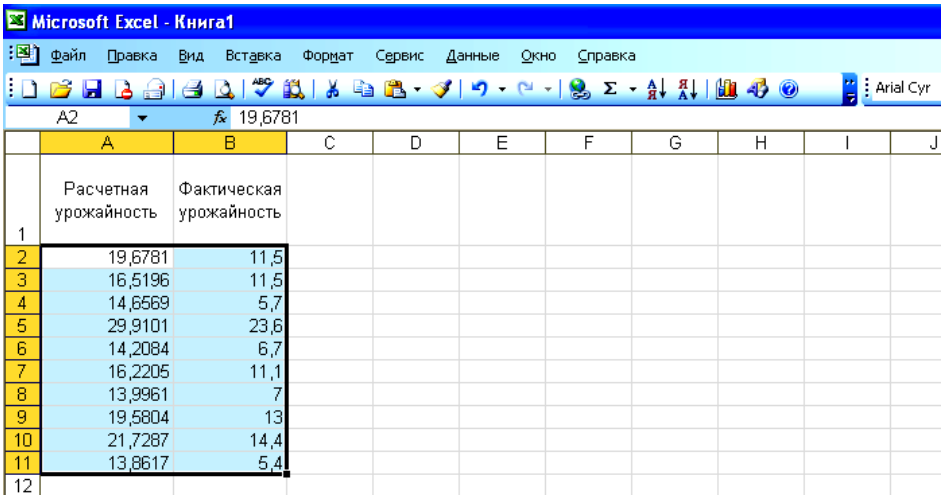


Рисунок 41 – Заполнение значений в MS Excel

Далее построим графики фактической и расчетной урожайности яровой пшеницы (рисунок 42).

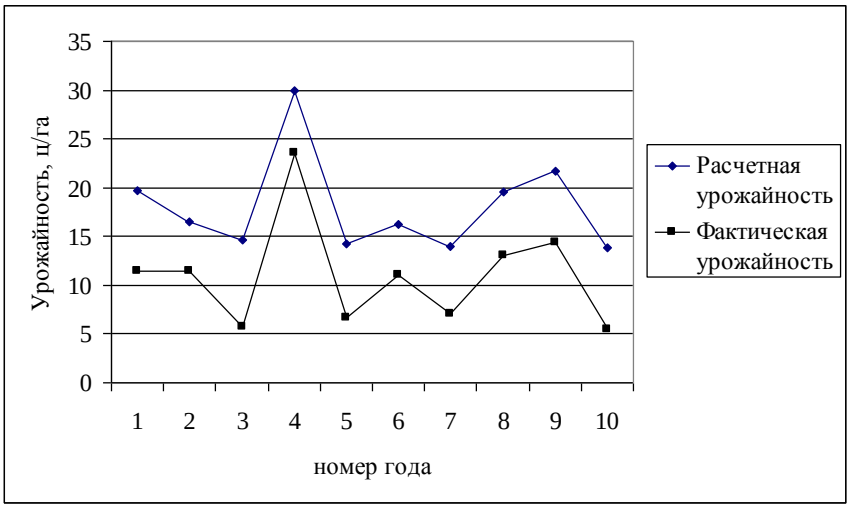


Рисунок 42 – Графики фактической и расчетной урожайности яровой пшеницы

Сравним фактическую и расчетную урожайность яровой пшеницы (таблица 4).

Таблица 14 – Сравнение динамических рядов фактической и расчетной урожайностей яровой пшеницы

№ года	Фактическая урожайность, ц/га	Расчетная урожайность, ц/га	Отклонение фактических значений от расчетных, ц/га
1.	11,5	19,68	-8,18
2.	11,5	16,52	-5,02
3.	5,7	14,66	-8,96
4.	23,6	29,91	-6,31
5.	6,7	14,21	-7,51
6.	11,1	16,22	-5,12
7.	7,0	13,86	-6,86
8.	13,0	19,58	-6,58
9.	14,4	21,73	-7,33
10.	5,4	14,00	-8,60

На протяжении анализируемого периода фактическая урожайность была меньше расчетной. Отклонение колеблется от 5,02 до 8,96 ц /га. Такое отклонение может быть вызвано некачественным посевным фондом или несоблюдением агротехнических сроков мероприятий. Снижение фактической урожайности также может быть связано с гибелью посевов в результате засухи или уничтожения вредителями. На снижение фактической урожайности немалое влияние может оказывать нехватка финансовых средств, а также неинтенсивные технологии производства.

Составим прогноз на основе рассмотрения парной зависимости между временным фактором и расчетной урожайностью. Для этого воспользуемся данными таблицы 4.15, вынесенными для удобства из таблицы 14.

Таблица 15 – Динамика расчетной урожайности яровой пшеницы

Номер года	Расчетная урожайность, ц/га
1.	19,68
2.	16,52
3.	14,66
4.	29,91
5.	14,21
6.	16,22
7.	13,86
8.	19,58
9.	21,73
10.	14,00

Для этого откроем новый файл и внесем данные из таблицы 4.15. Для этого в главном меню выбираем **Файл – Новый** (рисунок 43).

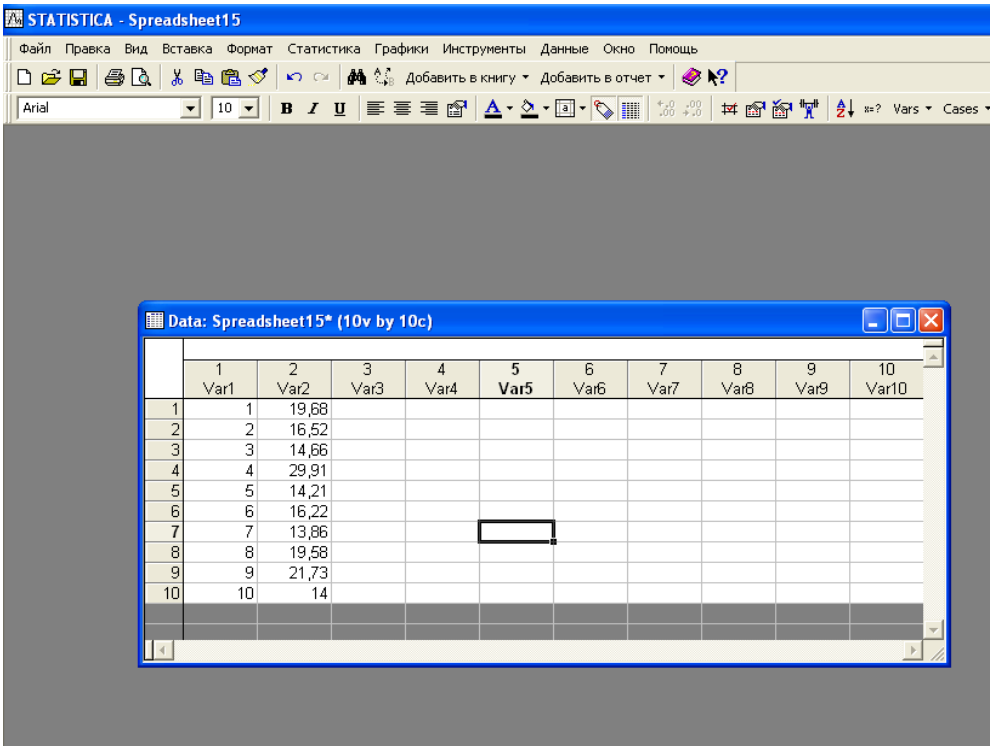


Рисунок 43 – Окно с заполненными данными

Затем среди команд выбираем **Статистика – Дополнительные Линейные/Нелинейные модели – Основные линейные модели** (рисунок 44).

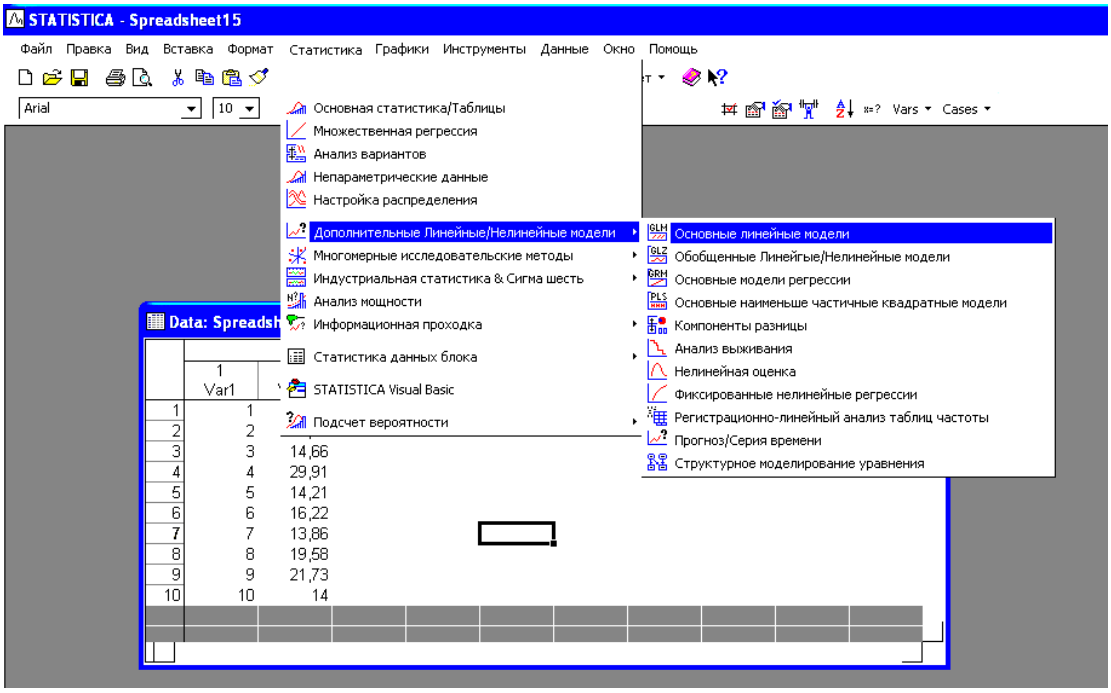


Рисунок 44 – Выбор основных линейных моделей

В появившемся окне **General Linear Models**, выбираем **Simple Regression**, затем **ОК** (рисунок 45).

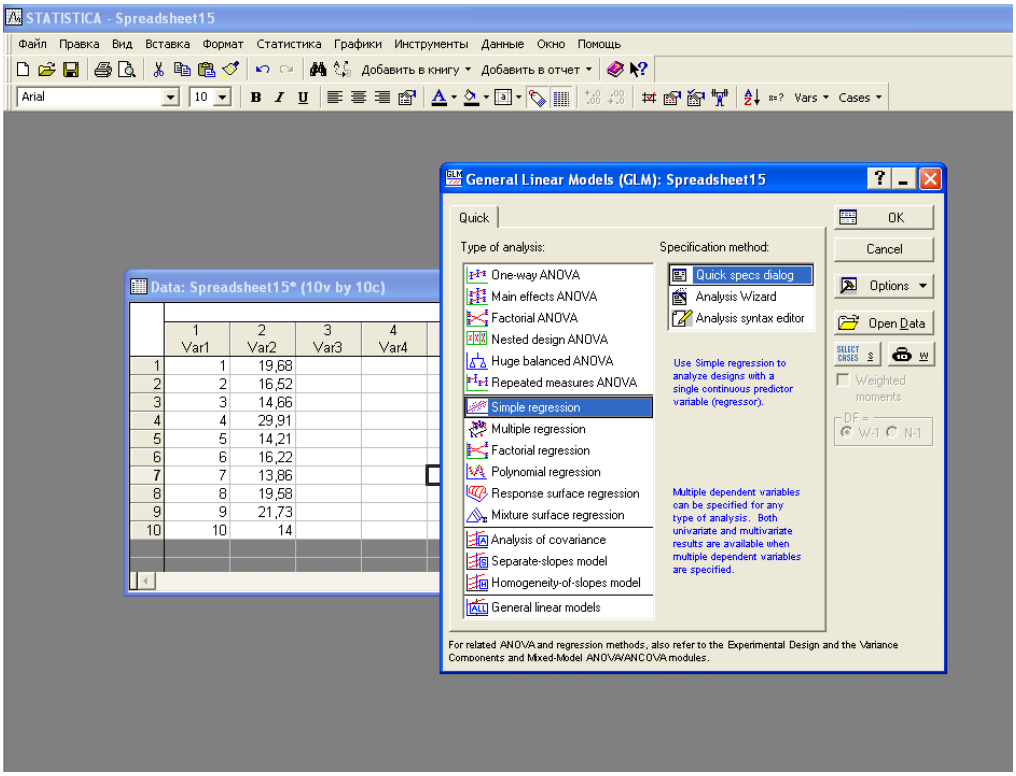


Рисунок 45 – Выбор метода

Затем нажимаем кнопку **Variables**. Выбираем зависимую (урожайность) и независимую (годы) переменные (рисунок 46).

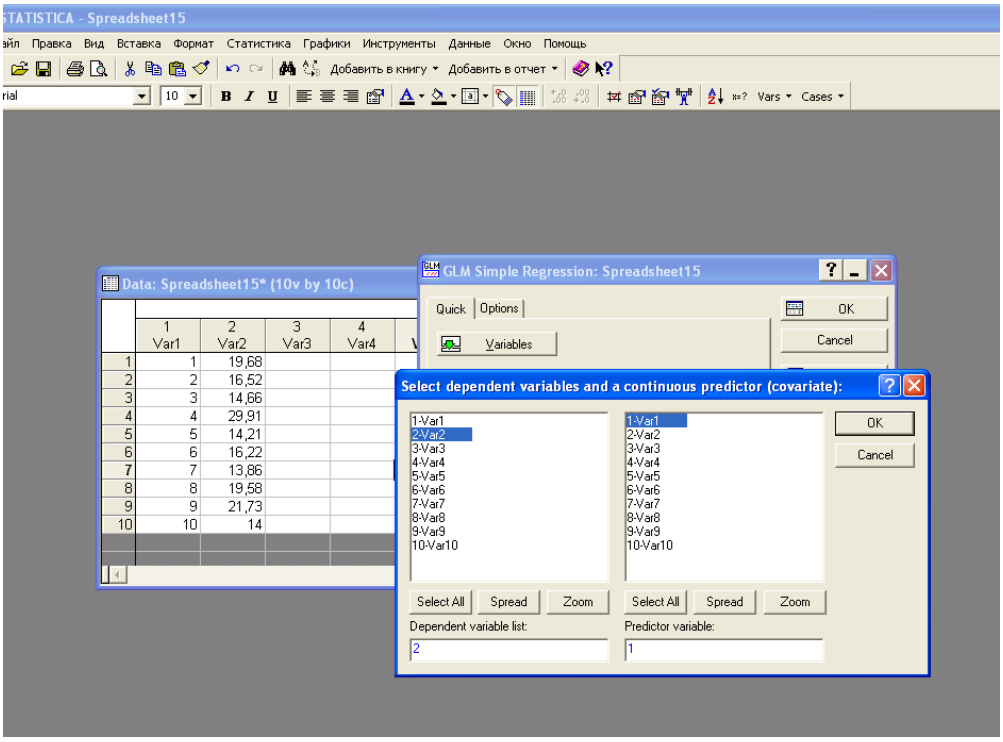


Рисунок 46 – Выбор переменных

Нажимаем **ОК**.

Затем в появившемся окне выбираем вкладку **Summary** и нажимаем кнопку **Coefficients** и **Whole model R** (рисунок 47).

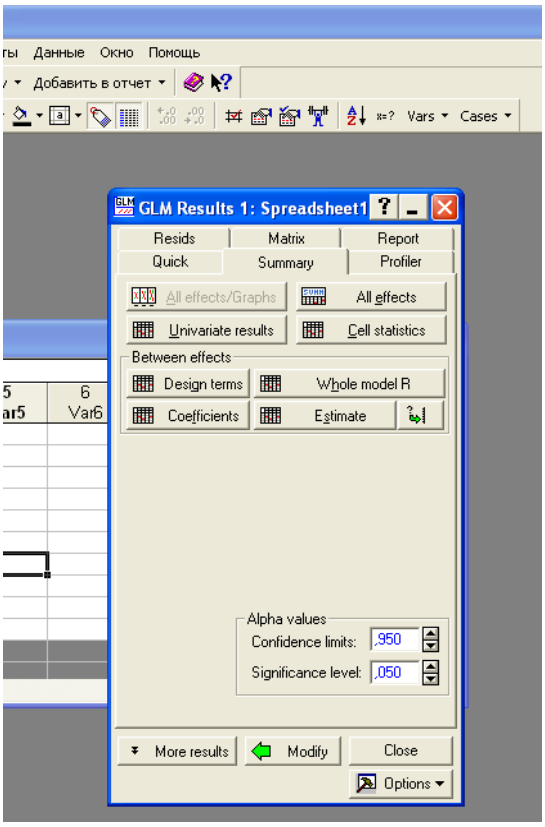


Рисунок 47 – Расчет коэффициентов

Появится окно, представленное на рисунке 48.

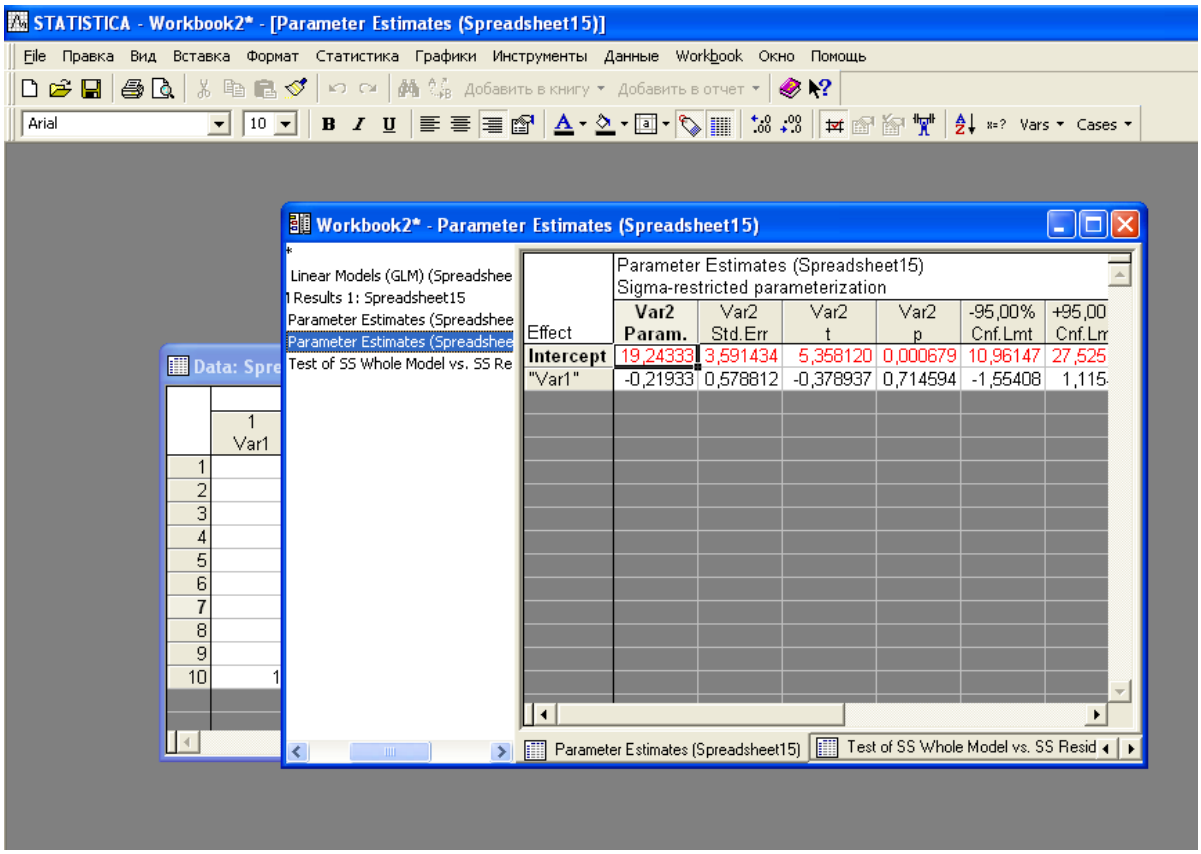


Рисунок 48 – Оценка параметров

Для наглядности представим данные в виде таблицы 16.

Таблица 16 – Параметры искомого уравнения парной линейной регрессии

Эффект	Параметр оценивает (Spreadsheet1) Sigma-ограниченная параметризация									
	урож Парамет	урож Std.Err	урож t	урож p	-95,00% Cnf.Lmt	+95,00% Cnf.Lmt	урож Beta (?)	урож St.Err.?	-95,00% Cnf.Lmt	+95,00% Cnf.Lmt
ОТРЕЗОК	19,24238	3,591740	5,357398	0,000680	10,95981	27,52495				
"Var1"	-0,21933	0,578861	-0,378904	0,714618	-1,55419	1,11552	-0,132777	0,350423	-0,940854	0,675300

Коэффициенты равны a_0 = 19,24238, a_1 =-0,21933.
Уравнение прямой имеет вид: y = 19,24238 – 0,21933x.

Вторая таблица в программе будет выглядеть так, как представлено на рисунке 49.

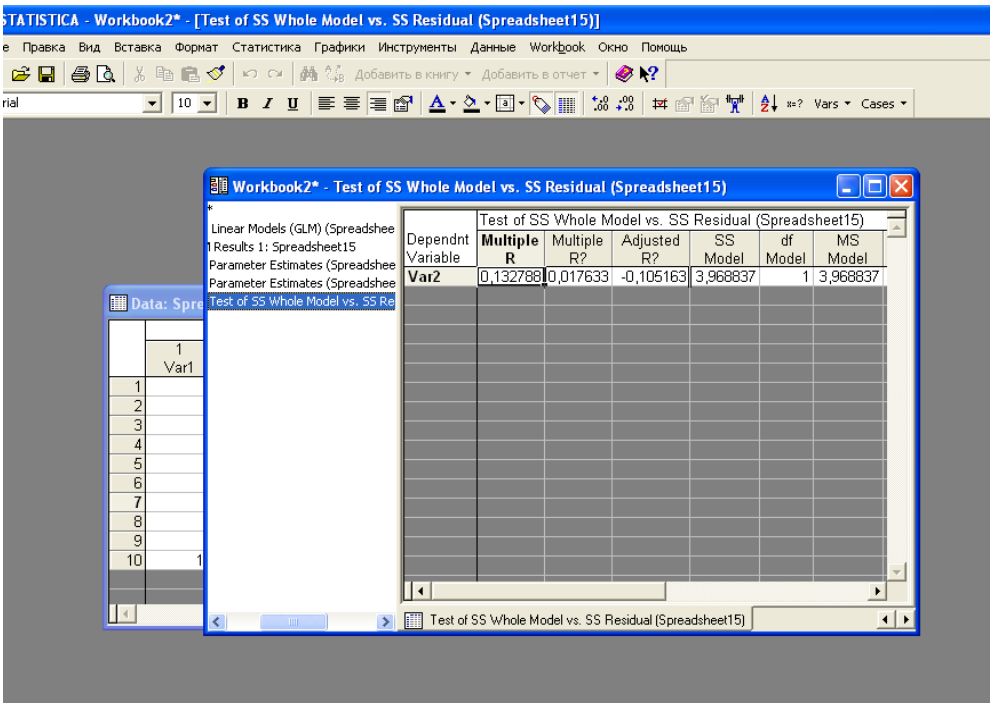


Рисунок 49 – Показатели адекватности и дисперсии

Более наглядно она представлена в таблице 17.

Таблица 17 – Показатели адекватности и дисперсии уравнения регрессии

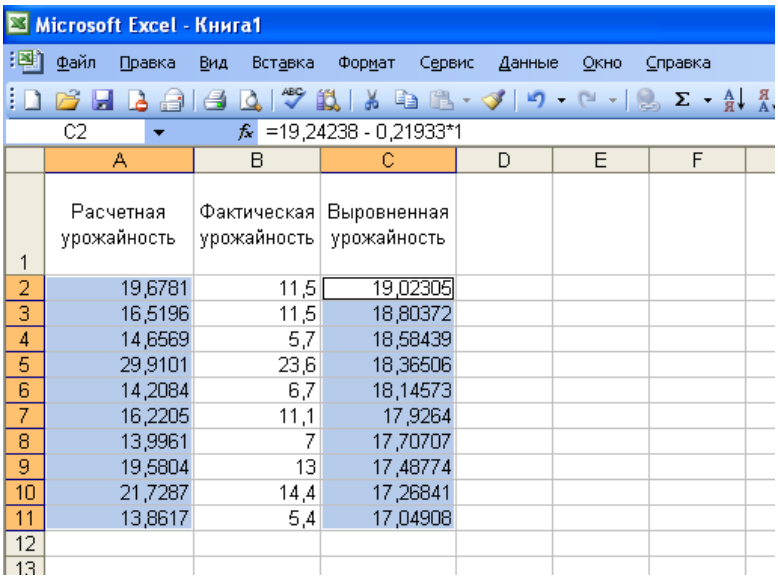
Dependnt Переменная	Test of SS Whole Model vs. SS Residual (Spreadsheet1)									
	Множеств R	Множеств R?	Назначен R?	SS Модель	df Модель	MS Модель	SS Резидент	df Резидент	MS Резидент	F	p
урож	0,132777	0,017630	-0,105167	3,968815	1	3,968815	221,1531	8	27,64413	0,143568	0,714618

Коэффициент корреляции = 0,132777
Multiple R? (коэффициент детерминации) = 0,017630
SS (Объясненная дисперсия) = 3,968815
MS (Остаточная дисперсия) = 3,968815
F-статистика Фишера = 0,143568
Поставим в полученное уравнение вместо x порядковые номера лет и получим выровненное значение расчетной урожайности яровой пшеницы за 10 лет:

y = 19,24238 – 0,21933x;

$y_1 = 19{,}24238 - 0{,}21933 \cdot 1 = 19{,}02305;$
$y_2 = 19{,}24238 - 0{,}21933 \cdot 2 = 18{,}80372;$
$y_3 = 19{,}24238 - 0{,}21933 \cdot 3 = 18{,}58439;$
$y_4 = 19{,}24238 - 0{,}21933 \cdot 4 = 18{,}36506;$
$y_5 = 19{,}24238 - 0{,}21933 \cdot 5 = 18{,}14573;$
$y_6 = 19{,}24238 - 0{,}21933 \cdot 6 = 17{,}9264;$
$y_7 = 19{,}24238 - 0{,}21933 \cdot 7 = 17{,}70707;$
$y_8 = 19{,}24238 - 0{,}21933 \cdot 8 = 17{,}48774;$
$y_9 = 19{,}24238 - 0{,}21933 \cdot 9 = 17{,}26841;$
$y_{10} = 19{,}24238 - 0{,}21933 \cdot 10 = 17{,}04908.$

Затем добавим столбец «Выровненная урожайность» в таблицу программы Microsoft Excel (рисунок 50).



	A	B	C	D	E	F
1	Расчетная урожайность	Фактическая урожайность	Выровненная урожайность			
2	19,6781	11,5	19,02305			
3	16,5196	11,5	18,80372			
4	14,6569	5,7	18,58439			
5	29,9101	23,6	18,36506			
6	14,2084	6,7	18,14573			
7	16,2205	11,1	17,9264			
8	13,9961	7	17,70707			
9	19,5804	13	17,48774			
10	21,7287	14,4	17,26841			
11	13,8617	5,4	17,04908			
12						
13						

Рисунок 50 – Подготовленные данные для построения графика

Построим графики расчетной и выровненной урожайности яровой пшеницы (рисунок 51).

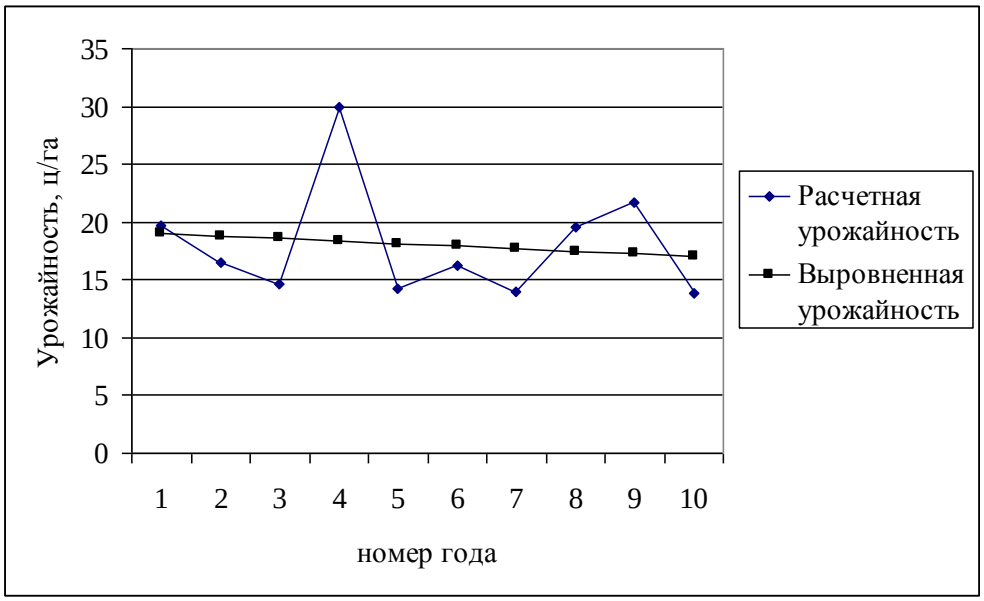


Рисунок 51 – Расчетная и выровненная урожайность яровой пшеницы

Данные расчетной и выровненной по прямой урожайности представлены в таблице 18.

Таблица 18 – Динамика расчетной и выровненной урожайности яровой пшеницы

Номер года	Расчетная урожайность, ц/га	Выровненная урожайность, ц/га
1	19,6781	19,02305
2	16,5196	18,80372
3	14,6569	18,58439
4	29,9101	18,36506
5	14,2084	18,14573
6	16,2205	17,9264
7	13,9961	17,70707
8	19,5804	17,48774
9	21,7287	17,26841
10	13,8617	17,04908

Так как прогнозирование можно осуществлять только на 1/3 от исследуемого периода, то далее составим прогноз урожайности на следующие 3 года. Для этого подставим в полученную формулу прямой номера лет с 11 по 13:

$y_{11} = 19{,}24238 - 0{,}21933 \cdot 11 = 16{,}82975;$

$y_{12} = 19{,}24238 - 0{,}21933 \cdot 12 = 16{,}61042;$

$y_{13} = 19{,}24238 - 0{,}21933 \cdot 13 = 16{,}39109.$

Затем нанесем на рисунок 51 график прогнозной урожайности (рисунок 52).

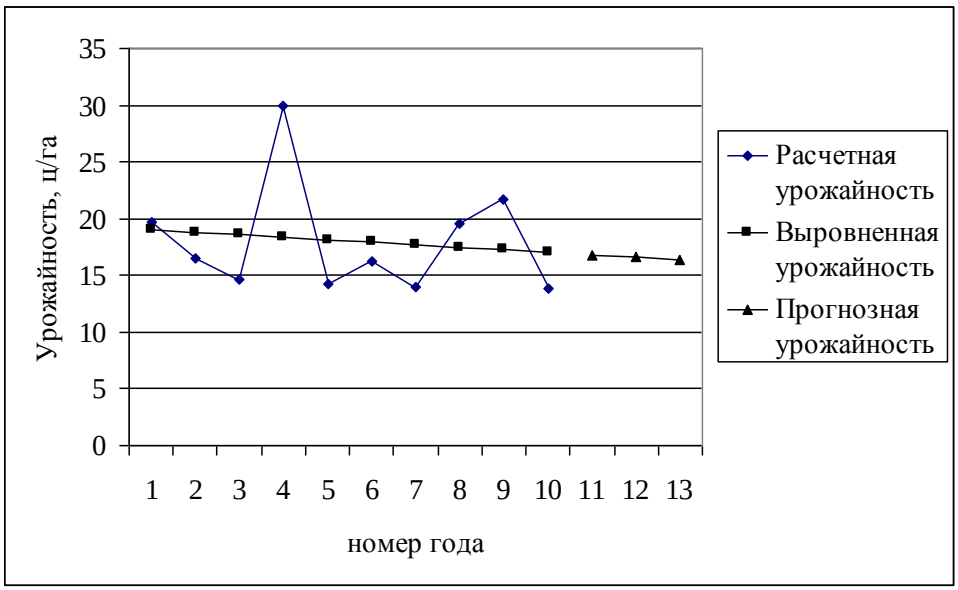


Рисунок 52 – Прогнозная урожайность

Таким образом, используя основные положения корреляционно-регрессионного моделирования, в частности, алгоритм множественной и парной корреляции, мы можем осуществлять прогнозирование на краткосрочный и среднесрочный период при условии экстраполяции полученного тренда, то есть сохранения условий и закономерностей прошлого развития исследуемого явления и процесса на будущее.

Исходные данные к задаче 1

№	prise	DEN	sherst	hlopok	firm
	Y	**X1**	**X2**	**X3**	**X4**
1	45	20	86	14	0
2	48	20	97	3	1
3	49	20	97	3	1
4	51	20	90	17	0
5	56	30	79	21	0
6	74	30	79	21	0
7	81	30	85	15	1
8	44	40	85	13	1
9	43	40	88	10	1
10	68	40	86	14	1
11	63	40	82	18	0
12	44	40	83	14	1
13	48	40	84	16	0
14	96	40	82	18	1
15	29	40	85	15	0
16	36	50	85	15	0
17	37	60	98	2	1
18	52	60	76	24	0
19	59	70	83	17	1
20	69	70	88	10	1
21	61	70	76	24	0
22	40	80	42	8	1
23	30	80	50	42	0
24	41	40	82	18	0
25	75	20	86	14	0
26	85	30	16	30	1
27	36	40	82	18	1
28	64	30	85	15	1
29	52	80	50	42	0
30	52	40	86	14	1
31	45	40	85	15	1
32	47	15	88	12	1
33	67	20	88	12	1
34	81	30	73	25	1
35	80	20	85	12	1
36	73	20	83	14	1
37	47	20	86	14	0
38	50	40	82	18	0
39	60	20	86	14	0
40	74	40	82	18	0
41	86	40	82	18	0
42	79	20	86	14	0
43	80	40	82	18	0
44	89	50	76	24	0
45	41	70	74	26	0

3. Выполнение индивидуальной расчетной работы.

3. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ

3.1 Практическое занятие 1, 2 (ПЗ-1, ПЗ-2) (4 часа).
Тема: «Обзорное итоговое занятие»

3.1.1 Задание для работы:

1. Подведение итогов.

3.1.2 Краткое описание проводимого занятия:

При подготовки к занятию необходимо обратить внимание на проработку всех вопросов, которые решались на предыдущих занятиях.

3.1.3 Результаты и выводы:

Студенты на обзорном итоговом занятии повторяют пройденный материал дисциплины, выполняют пропущенные лабораторные работы и иные виды деятельности для увеличения количества полученных за семестр обучения баллов рейтингового контроля.