Содержание:

Введение.

Применительно к системам баз данных архитектура клиент-сервер интересна и актуальна главным образом потому, что обеспечивает простое и относительно дешевое решение проблемы коллективного доступа к базам данных в локальной сети.

Реальное распространение архитектуры "клиент-сервер" стало возможным благодаря развитию и широкому внедрению в практику концепции открытых систем.

Основным смыслом подхода открытых систем является упрощение комплексирования вычислительных систем за счет международной и национальной стандартизации аппаратных и программных интерфейсов.

В основе широкого распространения локальных сетей компьютеров лежит известная идея разделения ресурсов. Высокая пропускная способность локальных сетей обеспечивает эффективный доступ из одного узла локальной сети к ресурсам, находящимся в других узлах.

Развитие этой идеи приводит к функциональному выделению компонентов сети: разумно иметь не только доступ к ресурсам удаленного компьютера, но также получать от этого компьютера некоторый сервис, который специфичен для ресурсов данного рода и программные средства. Так мы приходим к различению клиентов (рабочих станций) и серверов локальной сети.

Технология “клиент-сервер” применительно к СУБД сводится к разделению системы на две части – приложение-клиент и сервер базы данных. Эта архитектура совмещает лучшие черты обработки данных на мэйнфреймах и технологии файл-сервер. От мэйнфреймов технология “клиент-сервер” позаимствовала такие черты, как централизованное администрирование, безопасность, надежность. От технологии файл-сервер унаследованы низкая стоимость и возможность распределенной обработки данных, используя ресурсы компьютеров-клиентов.

Со времени возникновения архитектуры клиент-сервер появилось много вариантов архитектуры процессора БД, поскольку он во многом определяет успех всей системы.

В настоящее время можно считать, что бум технологий, связанных с клиент-серверной архитектурой, все еще продолжается - большинство работающих в настоящее время информационных систем выполнено в этой технологии. Однако актуальными являются направления, связанные с развитием этой идеи - так называемые трехслойные и многослойные, а также децентрализованные приложения.

Опыт последних лет разработки программного обеспечения (ПО)

показывает, что архитектура информационной системы должна выбираться с учетом нужд бизнеса, а не личных пристрастий разработчиков.

Не секрет, что правильная и четкая организация информационных бизнес-решений является слагающим фактором успеха любой компании. Особенно важным этот фактор является для предприятий среднего и малого бизнеса, которым необходима система, которая способна предоставить весь объем бизнес-логикидля решения задач компании. В то же время, такие системы для компаний со средним и малым масштабом сетей часто попадают под критерий —цена - качество, то есть должны обладать максимальной производительностью и надежностью при доступной цене.

2. Основные понятия.

Архитектура информационной системы - концепция, определяющая модель, структуру, выполняемые функции и взаимосвязь компонентов информационной системы.

Клиент-сервер (*Client-server*) — вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг (сервисов), называемыми серверами, и заказчиками услуг, называемыми клиентами.

Сервер — это программа, представляющая какие-то услуги другим программам и обслуживающая запросы клиентов на получение ресурсов

определенного вида.

Клиент — это программа, использующая услугу, представляемую программой сервера.

Часто люди клиентом или сервером просто называют компьютер, на

котором работает какая-либоиз этих программ. В сущности, клиент и сервер — это роли, исполняемые программами.

Клиенты и сервера физически могут находиться на одном компьютере. Одна и та же программа может быть и клиентом, и сервером одновременно.

2.1. История.

Основным недостатком персональных компьютеров является их невысокая вычислительная мощность и надежность, а также необходимость в приобретении дополнительных аппаратных средств для устранения изолированности отдельных персональных компьютеров друг от друга.

Как правило, пользователям компьютеров нужны и высокая вычислительная мощность и прекрасные свойства персональных компьютеров. Поэтому там, где для выполнения сложных вычислений используются мощные изолированные центральные компьютеры с терминалами, их пользователям периодически приходится ходить на персональные компьютеры для редактирования текстов или выполнения задач, использующих электронные таблицы. Это заставляет пользователей освоить 2 различные операционные системы (на больших машинах обычно установлены ОС MVS, VMS, VM, UNIX, а на персональных - MS DOS/MS Windows, OS/2 или Mac) и не решает задачи совместного использования данных.

В результате опроса представителей 300 крупнейших фирм США, использующих персональные компьютеры, выяснилось, что для 81% опрошенных необходим доступ к данным более чем одного компьютера.

Чтобы решить эту задачу, персональные компьютеры начали объединять в локальные сети и устанавливать на них специальные операционные системы,

например, NetWare фирмы Novell, для совместного использования компьютерами сети файлов, размещенных в различных узлах сети. Эта технология называется **файл-сервер.**

Однако файл-серверы имеют ряд недостатков. Они не позволяют в полной мере обеспечить конфиденциальность доступа и целостность данных. По сети файлы передаются целиком, независимо от того, какая часть содержащихся в них данных нужна пользователю. Это сильно перегружает сеть и уменьшает быстродействие системы. Невысока и надежность системы на основе файл-серверов. Сбой на одной из рабочих станций в момент записи файла приводит к потере или искажению данных.

Для обеспечения непротиворечивости данных приходится блокировать файлы, что также приводит к замедлению работы. Естественным желанием специалистов в области вычислительной техники было совместить преимущества персональных компьютеров и мощных центральных компьютеров.

Первым шагом в этом направлении явилось использование персональных компьютеров в качестве интеллектуальных терминалов. При таком подходе в персональном компьютере, соединенном с центральным компьютером, запускается специальное программное обеспечение, позволяющее этому персональному компьютеру работать в режиме эмуляции терминала. При этом мы получаем архитектуру, реализующую все достоинства архитектуры с мощным центральным компьютером, но, кроме того, персональный компьютер может использоваться и самостоятельно, по своему прямому назначению. Теперь нет необходимости иметь на столе 2

дисплея, однако большинство недостатков, присущих архитектуре с центральным компьютером, все еще сохраняется. Кроме того, хотя персональные компьютеры, имеющие дисплеи с картой VGA, позволяют работать с графикой, однако использовать их в качестве графического терминала большой центральной машины неудобно. Задача выполняется в центральном компьютере и по проводам передаются графические образы экрана. Эти образы довольно велики и скорость смены изображения на экране может быть очень низкой.

Следующим шагом в решении описанной выше проблемы явилось использование архитектуры клиент-сервер. В такой архитектуре все компьютеры сети разделены на 2 группы: клиенты и серверы. Компьютер-сервер - это мощный компьютер с большой оперативной памятью и большим количеством дискового пространства. На

нем хранится база данных и выполняется сложная обработка, требующая больших вычислительных ресурсов. На компьютерах-клиентах выполняются первичная обработка данных при вводе, форматирование данных, а также окончательная (финишная) обработка данных, извлеченных с сервера. В качестве компьютеров-клиентов обычно используются персональные компьютеры типа IBM PC или Macintosh. Преимущества архитектуры клиент-сервер очевидны. Каждый тип компьютера используется по своему назначению, а следовательно, обеспечивается более полное использование возможностей компьютеров.

На компьютерах-клиентах работают знакомые пользователям PC пакеты, позволяющие предоставлять результаты работы всей системы в удобном для анализа и принятия решений виде. На этих компьютерах легко можно реализовать дружественный пользовательский интерфейс приложения, использующий графику, цвет, звук, работу с окнами и мышью и т.д. Компьютер-клиент позволяет быстро выполнять ввод и первичный контроль данных. Для финишной обработки данных могут использоваться те редакторы или пакеты электронных таблиц, которые пользователь считает наиболее удобными. В качестве компьютеров-клиентов могут одновременно использоваться компьютеры разных типов с различными операционными системами.

Архитектура клиент-сервер позволяет реализовать распределенную обработку, поскольку часть работы (интерфейс с пользователем, финишная обработка) выполняется на компьютере-клиенте, а часть - на компьютере-сервере. Это позволяет снизить загрузку сервера и оптимизировать его работу, а также увеличить число клиентов, одновременно работающих с сервером.

Наиболее часто архитектура клиент-сервер применяется для приложений, созданных с использованием систем управления базами данных (СУБД).

Дальнейшим развитием архитектуры клиент-сервер явилось использование в сети не одного, а нескольких серверов баз данных. Это позволило перейти от работы с локальной БД к работе с распределенной БД.

Причем работа с распределенной базе данных (БД) "прозрачна" для пользователя, т.е. он работает с ней так же, как с локальной БД, не задумываясь о том, на каком сервере лежат его данные. Пользователь обращается к одному из серверов, тот, не найдя у себя нужных данных,

автоматически обращается к другим серверам.

Много серверная архитектура сегодня представляется очень перспективной. Она позволяет заменить одну мощную центральную машину на несколько менее мощных и, следовательно, более дешевых, и еще больше распараллелить обработку данных. Кроме того, такая архитектура повышает надежность системы, поскольку при выходе из строя одного из серверов все приложения, работающие с данными других серверов, могут продолжать работу. При выходе из строя части локальной или глобальной сети система может попытаться найти альтернативный путь к нужному серверу (по другим ветвям сети). Кроме того, на локальных серверах могут храниться данные, наиболее часто используемые в данном узле, что позволяет свести к минимуму передачу данных по сети от сервера к серверу.

3. Архитектура клиент-сервер как концепция локальной сети.

Клиент-сервер – технология, разделяющая СУБД на две части: клиентскую и серверную. На клиентской части формируются запросы к серверу и приходят результаты этих запросов для просмотра и дальнейшего использования, т.е. происходит «контакт с внешним миром». На компьютере-сервере расположены общие для всех клиентов данные и работает специальная программа - сервер баз данных, оптимизирующая выполнение запросов клиентов.

Главная мысль, заложенная в эту технологию - минимизировать объем данных, передаваемых по сети, поскольку основные потери времени и сбои происходят именно из-за недостаточно высокой пропускной способности сети.

Двухуровневая система клиент-сервер это:

Клиент - программа обработки, пользовательская и прикладная программа. Занимается обычно интерфейсом с пользователем, а всю фактическую работу с базой данных осуществляет сервер базы данных.

Сервер базы данных – компьютер или программа, предназначенные для обработки запросов от программ-клиентов. Серверы обычно обеспечивают работу сетевых служб, но иногда могут использоваться и в рамках одного компьютера. В отличие от обычных программ, которые запускаются, выполняют определенное задание и заканчивают работу, программа-сервер запускается и находится в пассивном состоянии ожидания запроса. Обработав поступивший запрос, сервер ожидает поступление следующего. Основное требование к серверу БД – обеспечение

минимального времени выполнения запросов при максимально возможном числе пользователей.

Технология клиент-сервер применяется, когда размеры баз данных велики, когда велики размеры вычислительной сети, и производительность при обработке данных, хранящихся не на компьютере пользователя. Если данная технология не применяется, то для обработки даже нескольких записей весь файл копируется на компьютер пользователя и только затем обрабатывается. При этом резко возрастает загрузка сети, и снижается производительность труда многих сотрудников.

4. Преимущества архитектуры клиент-сервер.

Несомненным преимуществом является приближенность данных к процессам вычисления. Практически, все расчеты выполняются на сервере, что увеличивает быстродействие в десятки и сотни раз.

В большинстве случаев программа обработки (клиентская часть) расположена на одном компьютере, а сама база данных хранится на другом. Помимо хранения централизованной базы данных центральная машина (сервер базы данных) обеспечивает выполнение основного объема обработки данных. Запрос на данные, выдаваемый клиентом (рабочей станцией), порождает поиск и извлечение данных на сервере. Извлеченные данные транспортируются по сети от сервера к клиенту. Причем, по сети передается только полезная информация.

Концепция условно изображается так:

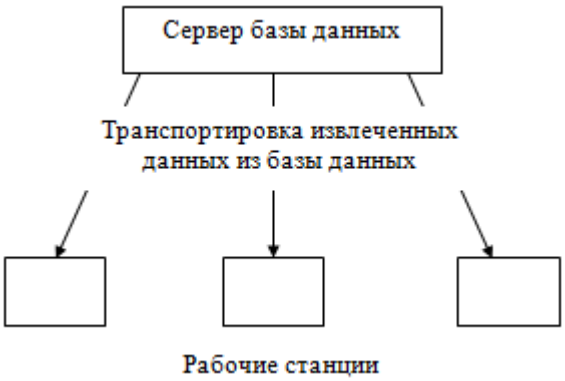

Рис. 1. Схема транспортировки данных.

Также преимущество архитектуры в том, что постоянно идет работа по совершенствованию самого метода хранения и обработки информации, и если его реализация (т.е.сервер базы данных) сменилась, то не потребуется перекомпилировать с новыми библиотеками все разработанные программы, а достаточно будет инсталлировать новый сервер базы данных взамен старого и перевести базы данных в формат нового сервера, применив для этого прилагаемую к нему утилиту.Так можно сделать, если новый сервер придерживается тех же правил обмена между ним и программой пользователя, что и старый.

Используя множество небольших компьютеров, разработчики систем клиент-сервер могут эмулировать вычислительную мощность больших ЭВМ, распределяя прикладную задачу по различным микрокомпьютерам и серверам. Каждый из них берет на себя свою часть вычислительной нагрузки, используя информацию совместно с другими процессорами сети. Суть идеи в том, чтобы повысить мощность системы, не увеличивая производительность одного компьютера, а суммируя средства многих.

Быстродействие - основной фактор целесообразности разработки систем для архитектуры клиент-сервер . Применение средств быстрой разработки программ (Rapid Application Development - RAD) позволяет разработчикам создавать прикладные системы для архитектуры клиент-сервер в рекордно короткие сроки. Технология серверов баз данных также становится проще в использовании и сочетается в одних системах со средствами RAD. Таким образом, сокращается время, необходимое для подготовки и передачи прикладной программы пользователю.

Спецификой архитектуры клиент-сервер является использование языка запросов SQL.

SQL(Structured Query Language – язык структурированных запросов) – универсальный язык, предназначенный для создания и выполнения запросов, обработки данных как в собственной базе данных приложения, так и с базами данных, созданных другими приложениями, поддерживающими SQL.

Microsoft Access, Microsoft Visual FoxPro, Microsoft Visual Basic обеспечивают средства для создания клиентских частей в приложениях клиент-сервер, которые сочетают в себе средства просмотра, графический интерфейс и средства построения запросов, а Microsoft SQL Server является на сегодняшний день одним из самых мощных серверов баз данных.

5. Компоненты архитектуры клиент-сервер.

Существуют три основных программных компонента архитектуры клиент-сервер :

- Программное обеспечение конечного пользователя.
- Промежуточное обеспечение.
- Программное обеспечение сервера.

1. К ПО конечного пользователя относятся средства разработки программ и генераторы отчетов, в том числе электронные таблицы и текстовые процессоры. С помощью этого ПО пользователи устанавливают связь с серверами, отправляют на рассмотрение серверу запросы и получают ответную информацию.
2. Промежуточное обеспечение предоставляет общий интерфейс для ПО конечного пользователя и сервера, проникающий сквозь сквозь слои GUI (графический интерфейс пользователя), операционную систему, вычислительной сети и собственных драйверов базы данных с помощью общих вызовов. Для завершения операции сервер базы данных выполняет запрос и передает клиенту затребованные данные для обработки их программой клиента.
3. Под ПО сервера подразумевается операционная система и конкретный сервер базы данных, используемый для обработки запросов клиентской части информационной системы.

Серверы баз данных занимаются не только обслуживанием данных. В них предусмотрены также механизмы блокировок и элементы управления многопользовательским доступом, которые обеспечивают защиту данных от опасности параллельного доступа. Кроме этого, серверу баз данных приходится ограждать данные от несанкционированного доступа, оптимизировать запросы к базе данных, обеспечивать кэширование и предоставлять место для размещения словаря данных.

Две другие важные особенности, на которые стоит обратить внимание, - способность сервера обеспечивать целостность ссылочных данных и обоюдный контроль завершения транзакции. *Ссылочная целостность данных* – это механизм, обеспечивающий каждому внешнему ключу соответствующий первичный ключ. *Обоюдный контроль завершения транзакций* - это гарантия того, что данные не будут повреждены даже при аппаратном сбое.

6. Модель клиент-сервер как основа построения информационных сервисов Интернет.

В основу взаимодействия компонентов информационных сервисов Сети в большинстве случаев положена модель клиент-сервер. Как правило, в качестве клиента выступает программа, которая установлена на компьютере пользователя, а в качестве сервера – программа, установленная у провайдера. В данном контексте под провайдером понимают организацию или частное лицо, которые поддерживают информационные ресурсы.

При этом возможны два варианта организации самой информационной системы, которая обеспечивает доступ к информационному ресурсу. Большинство систем Интернет построены по принципу взаимодействия «каждый с каждым», например, система World Wide Web, т.е. каждый пользователь может напрямую взаимодействовать с каждым сервером без посредников. Такой подход позволяет упростить всю технологическую схему построения системы, однако, приводит к порождению большого трафика в Сети. Альтернативный вариант построения системы, например, системы Usenet, когда пользователь может взаимодействовать только со «своим» сервером и не может обратиться к произвольному серверу в Сети. Однако, доступ он получает ко всей информации, которая присутствует в данной информационной системе, т.к. серверы обмениваются ею между собой.

Несколько таких схем показано на рисунке:

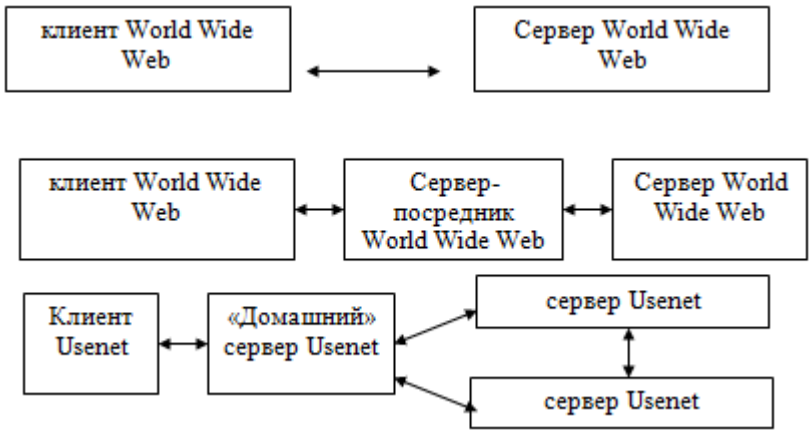

Рис. 2. Различные схемы клиент-сервер для информационных серверов Интернет

Принципиальным различием между схемой с посредником и схемой Usenet является то, что при посреднике работа по доступу к ресурсу перекладывается на его плечи. При этом он будет устанавливать соединение с каждым сервером в Сети. По схеме Usenet это делать не обязательно, т.к. информацию в принципе можно получить с любого сервера.

Заключение.

Архитектура клиент-сервер значительно упрощает и ускоряет разработку приложений за счет того, что правила проверки целостности данных находятся на сервере. Неправильно работающее клиентское приложение не может привести к потере или искажению данных. Все эти возможности, ранее свойственные только сложным и дорогостоящим системам, сейчас доступны даже небольшим организациям. Стоимость оборудования, программного обеспечения и обслуживания для персональных компьютеров в десятки раз ниже, чем для мэйнфреймов.

Реальное распространение архитектуры клиент-сервер стало возможным благодаря развитию и широкому внедрению в практику концепции открытых систем.

Основной проблемой систем, основанных на архитектуре клиент-сервер, является то, что в соответствии с этой концепцией от них требуется мобильность в как можно более широком классе аппаратно-программных решений открытых систем.

В заключение стоит отметить что архитектура клиент-сервер предоставляет разработчикам ПО исключительную свободу выбора и согласования различных типов компонентов для клиента, сервера и всех промежуточных звеньев.

Практическая часть.

В практической части моей курсовой работы требуется, используя ППП на ПК, рассчитать оптимальное сочетание цены и количества произведенного товара при максимальном значении получаемой прибыли путем задания переменных издержек на единицу товара. Наибольшую прибыль обеспечивают такой объем выпуска и цена, при которых предельные издержки максимально приближены к предельной выручке или равны ей. По данным таблицы нужно построить гистограмму с

заголовком, названием осей координат и легендой.

Решение задачи будет производиться в среде табличного процессора Microsoft Excel.

Описание алгоритма решения задачи можно представить в виде инфологической модели.

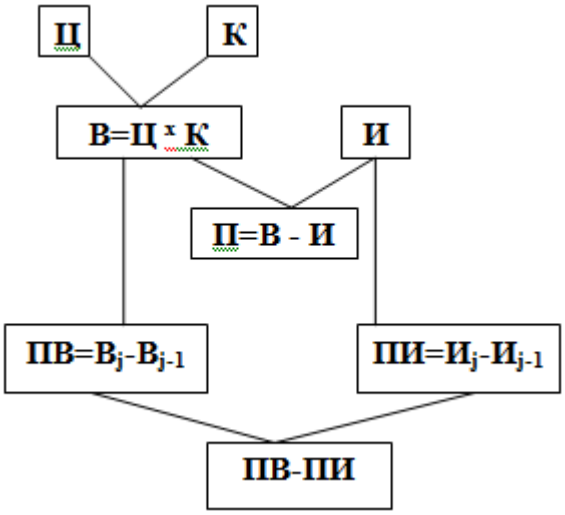

Рис. 3. Инфологическая модель решения задачи

Ц – цена товара

К – количество товара

И – суммарные издержки

В – выручка от реализации

П – прибыль

$\mathbf{B_j}$ – текущая выручка от реализации

$\mathbf{B_{j-1}}$ – предыдущая выручка от реализации

ПВ – предельная выручка

$\mathbf{И_j}$ – текущие суммарные издержки

$\mathbf{И_{j-1}}$ – предыдущие суммарные издержки

ПИ – предельные издержки

ПВ-ПИ – наибольшая прибыль

При выборе программного обеспечения я остановилась на использовании Microsoft Excel для решения экономической задачи. Вся задача заключается в вычислении одной таблицы, следовательно, использовать Microsoft Access нецелесообразно, т.к. для вычисления нужно использовать построитель выражений, который эффективно работает только при построении запросов.

Проектирование форм выходных документов и графическое представление данных по выбранной задаче

Таблицу с исходными данными на рабочем листе Microsoft Excel переношу в Microsoft Word.

Рис. 4. Таблица «Оптимальное сочетание цены и количества произведенного товара при максимальном значении получаемой прибыли» с исходными данными

Создаю структуру шаблона таблицы

Колонка электронной таблицы	Наименование (реквизит)	Тип данных	Формат данных	
			длина	точность
A	№ п/п	числовой	2	
B	Цена тыс. руб. (Ц)	числовой	5	3
C	Количество (К)	числовой	4	
D	Суммарные издержки, тыс. руб. (И)	числовой	3	

E	Выручка от реализации В=Ц * К	числовой	4
F	Прибыль П=В-И	числовой	3
G	Предельная выручка ПВ=Bj-Bj-1	числовой	2
H	Предельные издержки ПИ=Иj-Иj-1	числовой	2
I	ПВ-ПИ	числовой	1

Рис. 5. Структура шаблона таблицы «Оптимальное сочетание цены и количества произведенного товара при максимальном значении получаемой прибыли»

Создаю шаблон таблицы

Выручка от реализации В=Ц * К	**Прибыль П=В-И**	**Предельная выручка ПВ=Bj-Bj-1**	**Предельные издержки ПИ=Иj-Иj-1**	**ПВ-ПИ**
E4=B4*C4	F4=E4-D4	x	x	x
E5=B5*C5	F5=E5-D5	G5 =E5-E4	H5 =D5-D4	I5=G5-H5

E6=B6*C6	F6=E6-D6	G6 =E6-E5	H6 =D6-D5	I6 =G6-H6
E7=B7*C7	F7=E7-D7	G7 =E7-E6	H7 =D7-D6	I7 =G7-H7
E8=B8*C8	F8=E8-D8	G8 =E8-E7	H8 =D8-D7	I8 =G8-H8
E9=B9*C9	F9=E9-D9	G9 =E9-E8	H9 =D9-D8	I9 =G9-H9
E10=B10*C10	F10=E10-D10	G10 =E10-E9	H10 =D10-D9	I10 =G10-H10
E11=B11*C11	F11=E11-D11	G11 =E11-E10	H11 =D11-D10	I11 =G11-H11
E12=B12*C12	F12=E12-D12	G12 =E12-E11	H12 =D12-D11	I12 =G12-H12
E13=B13*C13	F13=E13-D13	G13 =E13-E12	H13 =D13-D12	I13 =G13-H13
E14=B14*C14	F14=E14-D14	G14 =E14-E13	H14 =D14-D13	I14 =G14-H14

E15=B15*C15	F15=E15-D15	G15 =E15-E14	H15 =D15-D14	I15 =G15-H15

Рис. 6. Шаблон таблицы «Оптимальное сочетание цены и количества произведенного товара при максимальном значении получаемой прибыли»

После подсчетов итоговую таблицу переношу из Microsoft Excel в Microsoft Word.

Рис. 7. Таблица «Оптимальное сочетание цены и количества произведенного товара при максимальном значении получаемой прибыли» с итоговыми данными
Представим результаты вычислений в графическом виде

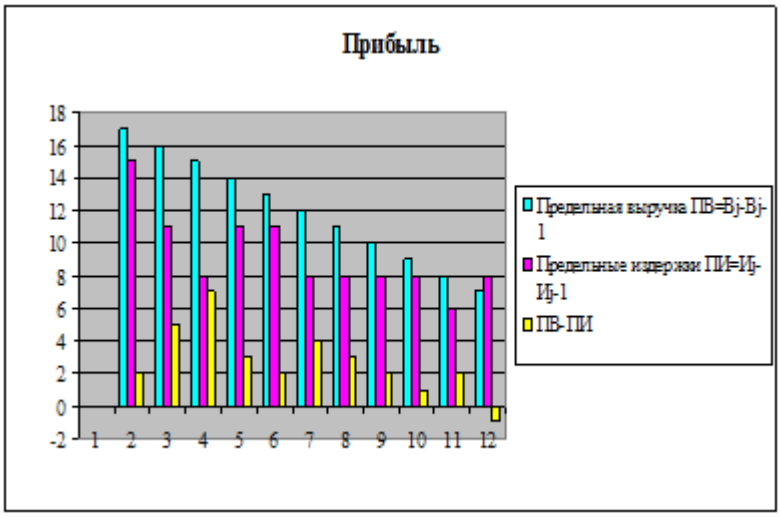

Рис. 8. Гистограмма «Прибыль»

Инструкция пользователя

1. Открываем табличный процессор Microsoft Excel.
2. Создаем таблицу «Оптимальное сочетание цены и количества произведенного товара при максимальном значении получаемой прибыли». Заголовок пишу в ячейке А1. Т.к. таблица имеет девять столбцов, то нужно объединить ячейки от А1 до I1, используя кнопку на панели инструментов «объединить и поместить в центре».
3. В ячейках с адресами от А3 до I3 пишем названия колонок таблицы, которые даны в методическом пособии. Чтобы задать ячейке размер необходимо поставить курсор мыши на правую границу ячейки и щелкнуть два раза.

4. Заполняем колонки данными, предложенными в методическом пособии. Тип данных в каждой колонке числовой.
5. Необходимо ввести значение даты между таблицей и ее названием. Для этого нужно объединить ячейки от A2 до I2, используя кнопку на панели инструментов «объединить и поместить в центре».
6. Для того чтобы произвести вычисления в таблице, я активизирую ячейку E4, ставлю знак =, активизирую ячейку B4, ставлю знак *, активизирую ячейку C4 и нажимаю клавишу Enter. В строке формул видим: E4=B4*C4.
7. Аналогично проводятся вычисления для других ячеек столбца «Выручка от реализации».
8. Для более быстрого вычисления элементов ячеек используем автозаполнение. Для этого нужно поставить курсор мыши на нижний правый угол активизированной ячейки и довести его до конца данного столбца (т.е. копировать ячейки). Таким способом можно производить расчеты для ячеек всех столбцов, а также заполнять ячейки уже предложенными данными, где числа изменяются на одинаковую величину.
9. После произведения всех необходимых расчетов требуется построить гистограмму. Для этого на панели инструментов выбираем кнопку «Мастер диаграмм».
10. В открытом окне «Мастер диаграмм» выбираем нужный тип, в данном случае гистограмму. Нажимаем кнопку «Далее».
11. Для создания гистограммы нужно щелкнуть в поле «Диапазон». Затем указать на листе ячейки, содержащие необходимые для построения данные.Нажимаем кнопку «Далее».
12. Для создания легенды нужно в поле «Легенда» указать значок «Добавить легенду» и ее размещение (например, справа). Нажимаем кнопку «Далее».
13. Затем указываем размещение гистограммы. В данном случае надо отметить «размещение на имеющемся листе». Нажимаем кнопку «Готово».

Задачи подобного типа можно решать вручную и используя вышеописанный проект. Но при вычислении вручную я затрачу больше времени, чем по своему разработанному программному решению. Также при преобразованиях вручную существует вероятность большего совершения ошибок. Мой проект обеспечивает простоту, быстроту и точность вычислений. Он является работоспособным, и по нему можно решать задачи подобного типа.

Список литературы.

1. Агальцов, В.П. Информатика для экономистов: Учебник / В.П. Агальцов, В.М. Титов. - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. - 448 с.
2. Божко, В.П. Информатика: данные, технология, маркетинг / В.П. Божко, В.В. Брага, Н.Г. Бубнова. - М.: Финансы и статистика, **2014**. - 224 с.
3. Бодров, О.А. Предметно-ориентированные экономические информационные системы: Учебник для вузов / О.А. Бодров. - М.: Гор. линия-Телеком, 2013. - 244 с.
4. Васильков, А.В. Информационные системы и их безопасность: Учебное пособие / А.В. Васильков, А.А. Васильков, И.А. Васильков. - М.: Форум, 2013. - 528 с.
5. Гейн, А.Г. Основы информатики и вычислительной техники / А.Г. Гейн, В.Г. Житомирский, Е.В. Линецкий, и др.. - М.: Просвещение, **2013**. - 254 с.
6. Горячев, А.В. Практикум по информационным технологиям / А.В. Горячев, Ю.А. Шафрин. - М.: Бином, **2016**. - 272 с.
7. Информатика: Энциклопедический словарь для начинающих / ред. Д.А. Поспелов. - М.: Педагогика-Пресс, **2013**. - 352 с.
8. Кушниренко, А.Г. Основы информатики и вычислительной техники / А.Г. Кушниренко, Г.В. Лебедев, Р.А. Сворень. - Л.: Просвещение; Издание 3-е, **2013**. - 224 с.
9. Румянцева, Е.Л. Информационные технологии: Учебное пособие / Е.Л. Румянцева, В.В. Слюсарь; Под ред. Л.Г. Гагарина. - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2013. - 256 с.
10. Светлов, Н.М. Информационные технологии управления проектами: Учебное пособие / Н.М. Светлов, Г.Н. Светлова. - М.: НИЦ ИНФРА-М, 2012. - 232 с.
11. Федотова, Е.Л. Информационные технологии в профессиональной деятельности: Учебное пособие / Е.Л. Федотова. - М.: ИД ФОРУМ, НИЦ ИНФРА-М, 2012. - 368 с.
12. Хейлсберг, А. Язык программирования C#. Классика Computers Science / А. Хейлсберг, М. Торгерсен, С. Вилтамут. - СПб.: Питер, 2012. - 784 с.