АННОТАЦИЯ

В данной работе рассмотрено построение информационных систем с использованием нечетких баз данных и описаны общие понятия, виды и модели информационных систем и баз данных. Проведен обзор продуктов для разработки и администрирования баз данных. Также представлены способы построения нечетких запросов к базе данных. Исследованы такие программные пакеты и системы как Fuzi Calc, CubiCalc, Rule Maker.

ANNOTATION

The purpose of my work is to review the development of information systems using fuzzy databases and to describe the General concept, types and models of information systems and databases. Also in this diploma I reviewed products, that are used for development and administration of databases. Methods of building a fuzzy queries to the database have been described here as well. Also I investigated some Software Packages and systems like Fuzi Calc, CubiCalc, Rule Maker

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....7

1 ИНФОРМАЦИОННЫЕ СИСТЕМЫ И БАЗЫ ДАННЫХ.....9

1.1 Свойства и структура информационных систем.....9

1.2 Типы и классификация информационных систем.....11

1.3 Области применения и примеры реализации информационных систем…16

1.4 Общие понятия о базах данных.....19

1.5 Системы управления базами данных.....21

1.6 Классификация систем управления базами данных.....24

1.7 Структурные элементы базы данных.....29

1.8 Обзор продуктов для разработки и администрирования баз данных.....33

1.9 Нечеткие базы данных и запросы в них.....35

2 ПРИМЕНЕНИЕ НЕЧЕТКИХ БАЗ ДАННЫХ В ИНФОРМАЦИОННЫХ СИСТЕМАХ.....45

2.1 Нечеткая логика и нечеткие системы.....45

2.2 Электронная таблица Fuzi Calc.....48

2.3 Пакет CubiCalc. Управление динамическими системами.....52

2.4 Формализация данных и выявление правил. Программа Rule Maker для Cubi Calc 2.0.....59

ЗАКЛЮЧЕНИЕ.....62

СПИСОК ЛИТЕРАТУРЫ.....64

ПРИЛОЖЕНИЕ 1.....65

ПРИЛОЖЕНИЕ 2.....67

ВВЕДЕНИЕ

Одной из основных сфер использования компьютеров считается обработка и хранение крупных размеров информации в разных областях человеческой деятельности: в экономике, в банковском процессе, торговле, транспорте, медицине, науке и т.д.

Главные идеи информационных технологических процессов в настоящий момент основываются на концепции, в соответствии с которой сведения должны быть сформированы в базы данных с целью отражения меняющегося общества и удовлетворения информационной необходимости пользователей.

Имеющиеся в настоящий момент информационные системы характеризуются большими размерами данных, трудной организацией, потребностью исполнять разные запросы множества пользователей.

Информационная система осуществляет автоматическое получение, обработку и манипулирование данными и содержит технические средства обработки данных, программное обеспечение и обслуживающий персонал. Обработка данных об объектах окружающего мира является задачей каждой информационной системы, а база данных - основа этой системы.

Актуальность темы в том, что в настоящее время ни одна организация не обходится без баз данных, которая повышает оперативность работы с информацией. Каждый человек пользуется телефонной книгой в своем смартфоне, а это тоже своеобразная база данных. Для того, чтобы найти определенный номер можно пролистать всю телефонную книгу, а если там 500 записей, то уже найти запись будет сложно. Поэтому в базах данных существуют запросы, благодаря которым поиск упрощается.

В запросах к базам данных, которые формулирует человек, чаще всего присутствуют неточности и неопределенности. Поэтому нет ничего удивительного, когда на запрос в Интернете пользователю выдается множество ссылок, упорядоченных по степени релевантности (или

соответствия) запросу. Потому что текстовой информации изначально предполагает нечеткость и неопределенность.

Например, нужно найти не что-то конкретное, а несколько записей, удовлетворяющих запросу. Так при поиске квартиры в Москве, можно задать запрос: «Найти предложения о сдаче не очень дорогого жилья близко к метро», для этого нужно использовать нечеткий запрос.

1 ИНФОРМАЦИОННЫЕ СИСТЕМЫ И БАЗЫ ДАННЫХ

1.1 Свойства и структура информационных систем

Информационная система – это взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели.

Современное понимание информационной системы предполагает использование в качестве основного технического средства переработки информации – компьютер. Кроме того, техническое применение информационной системы само по себе ничего не будет значить, если не учтена роль человека, для которого предназначена необходимая информация и без которого невозможно ее получение и представление.

Необходимо понимать разницу между компьютерами и информационными системами. Компьютеры, оснащенные специализированными программными средствами, являются технической базой и инструментом для информационных систем. Информационная система немыслима без персонала, взаимодействующего с компьютерами и телекоммуникациями [1].



Рисунок 1.1 – Свойства информационных систем

Общую структуру информационных систем (рис. 1.2) можно рассматривать как совокупность подсистем независимо от сферы применения.



Рисунок 1.2 – Общая структура информационных систем

Внедрение информационных систем может способствовать:

- получение более рациональных вариантов решения управленческих задач за счет внедрения математических методов;
- освобождение работников от рутинной работы за счет ее автоматизации;
- обеспечение достоверности информации;
- совершенствование структуры информационных потоков (включая систему документооборота);
- предоставление потребителям уникальных услуг;
- уменьшение затрат на производство продуктов и услуг (включая информационные).

Важнейшими принципами построения эффективных информационных систем являются:

- принцип интеграции, заключающийся в том, что обрабатываемые данные, однажды введенные в систему, многократно используются для решения большого числа задач.

- принцип системности, заключающийся в обработке данных в различных аспектах, чтобы получить информацию, необходимую для принятия решений на всех уровнях управления.

- принцип комплексности, заключающийся в механизации и автоматизации процедур преобразования данных на всех этапах функционирования информационной системы.

1.2 Типы и классификация информационных систем

Тип информационной системы зависит от того, чьи интересы она обслуживает и на каком уровне управления. Полная классификация информационных систем расположена в Приложении II [1].

Фактографические информационные системы накапливают и хранят данные в виде множества экземпляров одного или нескольких типов структурных элементов (информационных объектов). Каждый из таких экземпляров или некоторая их совокупность отражают сведения по какому-либо факту, событию отдельно от всех прочих сведений и фактов. Структура каждого типа информационного объекта состоит из конечного набора реквизитов, отражающих основные аспекты и характеристики объектов данной предметной области. Комплектование информационной базы в фактографических информационных системах включает, как правило, обязательный процесс структуризации входной информации. Фактографические информационные системы предполагают удовлетворение информационных потребностей непосредственно, т.е. путем представления потребителям самих сведений (данных, фактов, концепций).

В документальных (документированных) информационных системах единичным элементом информации является нерасчлененный на более мелкие элементы документ и информация при вводе (входной документ), как

правило, не структурируется, или структурируется в ограниченном виде. Для вводимого документа могут устанавливаться некоторые формализованные позиции (дата изготовления, исполнитель, тематика). Некоторые виды документальных информационных систем обеспечивают установление логической взаимосвязи вводимых документов. Определение и установление такой взаимосвязи представляет собой сложную многокритериальную и многоаспектную аналитическую задачу, которая не может быть формализована в полном объеме.

В геоинформационных системах данные организованы в виде отдельных информационных объектов (с определенным набором реквизитов), привязанных к общей электронной топографической основе (электронной карте). Геоинформационные системы применяются для информационного обеспечения в тех предметных областях, структура информационных объектов и процессов в которых имеет пространственно-географический компонент (маршруты транспорта, коммунальное хозяйство и т.д.).

Функциональный признак определяет назначение подсистемы, а также ее основные цели, задачи и функции. На рисунке 1.3 представлена классификация информационных систем по характеристике их функциональных подсистем.

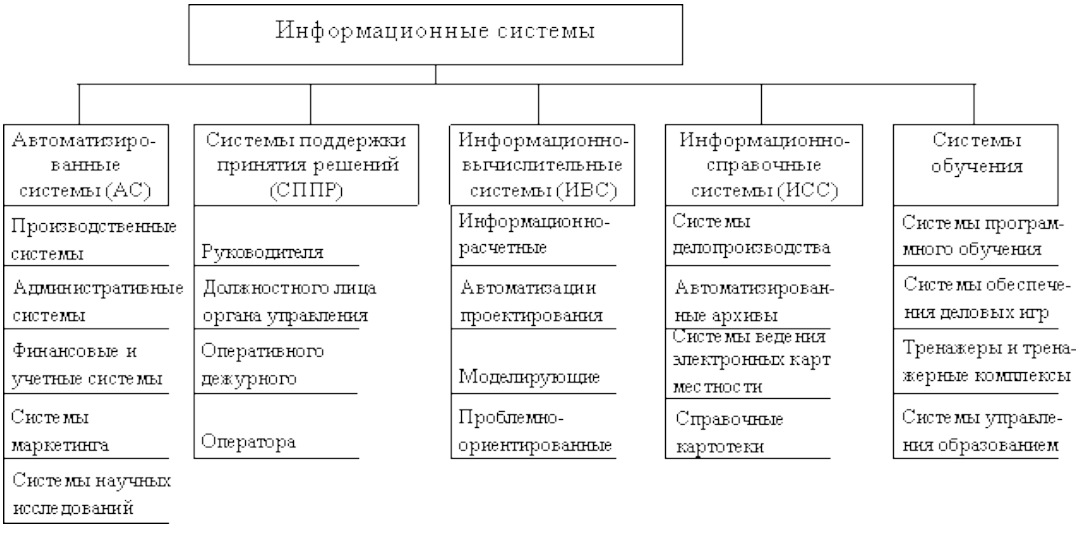

Рисунок 1.3 – Классификация ИС по функциональному признаку

В хозяйственной практике производственных и коммерческих объектов типовыми видами деятельности, которые определяют функциональный признак классификации информационных систем, являются производственная, маркетинговая, финансовая, кадровая деятельность.

Также выделяют информационные системы по уровням управления.

Информационная система оперативного уровня поддерживает специалистов-исполнителей, обрабатывая данные о сделках и событиях (счета, накладные, зарплата, кредиты, поток сырья и материалов). Назначение информационной системы на этом уровне - отвечать на запросы о текущем состоянии и отслеживать поток сделок в фирме, что соответствует оперативному управлению. Чтобы с этим справляться, информационная система должна быть легко доступной, непрерывно действующей и предоставлять точную информацию.

Задачи, цели и источники информации на оперативном уровне заранее определены и в высокой степени структурированы. Решение запрограммировано в соответствии с заданным алгоритмом. Информационная система оперативного уровня является связующим звеном между фирмой и внешней средой. Если система работает плохо, то организация либо не получает информации извне, либо не выдает информацию. Кроме того, система оперативного уровня – это основной поставщик информации для остальных типов информационных систем в организации, т.к. содержит и оперативную, и архивную информацию.

Информационные системы специалистов помогают им повышать продуктивность и производительность работы инженеров и проектировщиков. Задача подобных информационных систем - интеграция новых сведений в организацию и помощь в обработке бумажных документов. По мере того как индустриальное общество трансформируется в информационное, эффективность экономики все больше будет зависеть от уровня развития этих систем. Такие системы, особенно в виде рабочих станций и офисных систем, наиболее быстро развиваются сегодня в бизнесе.

Информационные системы офисной автоматизации вследствие своей простоты и многопрофильности активно используются работниками любого организационного уровня. Наиболее часто их применяют работники средней квалификации: бухгалтеры, секретари, клерки. Основная цель - обработка данных, повышение эффективности их работы и упрощение канцелярского труда. Информационные системы офисной автоматизации связывают воедино работников информационной сферы в разных регионах и помогают поддерживать связь с покупателями, заказчиками и другими организациями. Их деятельность в основном охватывает управление документацией, коммуникации, составление расписаний и т.д.

Эти системы выполняют следующие функции:

- обработка текстов на компьютерах с помощью различных текстовых процессоров;

- производство высококачественной печатной продукции;

- архивация документов;

- электронные календари и записные книжки для ведения деловой информации;

- электронная и аудиопочта;

- видео- и телеконференции.

Информационные системы обработки данных, в том числе и экспертные системы, вбирают в себя знания, необходимые инженерам, юристам, ученым при разработке или создании нового продукта. Их работа заключается в создании новой информации и нового знания. Так, например, существующие специализированные рабочие станции по инженерному и научному проектированию позволяют обеспечить высокий уровень технических разработок.

Основные функции информационных систем тактического уровня:

- сравнение текущих показателей с прошлыми показателями;

- составление периодических отчетов за определенное время (а не выдача отчетов по текущим событиям, как на оперативном уровне);

- обеспечение доступа к архивной информации и т.д.

Системы поддержки принятия решений обслуживают частично структурированные задачи, результаты которых трудно спрогнозировать заранее (имеют более мощный аналитический аппарат с несколькими моделями). Информацию получают из управленческих и операционных информационных систем. Используют эти системы все, кому необходимо принимать решение: менеджеры, специалисты, аналитики. Например, их рекомендации могут пригодиться при принятии решения покупать или взять оборудование в аренду.

Развитие и успех любой организации (фирмы) во многом определяются принятой в ней стратегией. Под стратегией понимается набор методов и средств решения перспективных долгосрочных задач. В этом контексте можно воспринимать и понятия стратегический метод, стратегическое средство, стратегическая система.

В настоящее время, в связи с переходом к рыночным отношениям, фирмы стали уделять большое внимание вопросам стратегии развития и поведения, что способствовало коренному изменению во взглядах на информационные системы. Они стали рассматриваться как стратегически важные системы, которые влияют на изменение выбора целей фирмы, ее задач, методов, продуктов, услуг, позволяя опередить конкурентов, а также наладить более тесное взаимодействие потребителей с поставщиками. Появился новый тип информационных систем - стратегический.

Стратегическая информационная система - компьютерная информационная система, обеспечивающая поддержку принятия решений по реализации перспективных стратегических целей развития организации. Известны случаи, когда новое качество информационных систем заставляло изменять не только структуру, но и профиль деятельности фирм, обеспечивая их процветание. Однако при этом возможно возникновение нежелательной психологической обстановки, связанное с автоматизацией некоторых

функций и видов работ, так как это может поставить некоторую часть работающих в затруднительное положение по вопросам занятости.

1.3 Области применения и примеры реализации информационных систем

В последние несколько лет компьютер стал неотъемлемой частью управленческой системы предприятий. Современный подход к управлению предполагает вложение денег в информационные технологии. Причем чем крупнее предприятие, тем больше должны быть подобные вложения.

Благодаря стремительному развитию информационных технологий наблюдается расширение области их применения. Если раньше чуть ли не единственной областью, в которой применялись информационные системы, была автоматизация бухгалтерского учета, то сейчас наблюдается внедрение информационных технологий во множество других областей. Эффективное использование корпоративных информационных систем позволяет делать более точные прогнозы, расчеты, аналитические сравнения и избегать возможных ошибок в управлении.

Из любых данных и отчетов о работе предприятия можно извлечь массу полезных сведений. И информационные системы как раз и позволяют извлекать максимум пользы из всей имеющейся в компании информации.

Именно этим фактом и объясняются жизнеспособность и бурное развитие информационных технологий – современный бизнес крайне чувствителен к ошибкам в управлении, и для принятия грамотного управленческого решения в условиях неопределенности и риска необходимо постоянно держать под контролем различные аспекты финансово-хозяйственной деятельности предприятия (независимо от профиля его деятельности).

Поэтому можно вполне обоснованно утверждать, что в жесткой конкурентной борьбе большие шансы на победу имеет предприятие, использующее в управлении современные информационные технологии.

Рассмотрим наиболее важные задачи, решаемые с помощью специальных программных средств.

Бухгалтерский учет – классическая и наиболее часто реализуемая на сегодняшний день область применения информационных технологий. Такое положение вполне объяснимо. Во-первых, ошибка бухгалтера может стоить очень дорого, поэтому очевидна выгода автоматизации бухгалтерии. Во-вторых, задача бухгалтерского учета довольно легко формализуется, так что разработка систем автоматизации бухгалтерского учета не представляет технически сложной проблемы.

Внедрение информационных технологий в управление финансовыми потоками также обусловлено критичностью этой области управления предприятия к ошибкам. Неправильно построив систему расчетов с поставщиками и потребителями, можно спровоцировать кризис наличности даже при налаженной сети закупки, сбыта и хорошем маркетинге. И наоборот, точно просчитанные и жестко контролируемые условия финансовых расчетов могут существенно увеличить оборотные средства фирмы [2].

Четкое и правильное управление производственным процессом представляет собой очень трудоемкую задачу. Основным механизмом здесь является планирование.

Автоматизация при решении подобной задачи дает возможность грамотно планировать, учитывать затраты, заниматься технической подготовкой производства, оперативно управлять процессом выпуска продукции в соответствии с технологией производственного процесса.

Очевидно, что чем крупнее производство, тем большее число бизнес-процессов участвует в создании прибыли, а значит, использование и внедрение информационных систем крайне необходимо.

Управление в работе маркетинга подразумевает сбор и анализ данных о фирмах-конкурентах, их предлагаемой продукции и ценовой политике, а также моделирование параметров внешнего окружения для определения

оптимального выгодного уровня цен, прогнозирования прибыли и планирования рекламных компаний. Для решения большинства этих задач, все данные могут быть формализованы и представлены в виде информационной системы, позволяющей существенно повысить эффективность маркетинга.

Документооборот является очень важным процессом деятельности любого предприятия. Хорошо отлаженная система учетного документооборота отражает реально происходящую на предприятии текущую производственную деятельность и дает управленцам большую возможность воздействовать и повышать эффективность управления.

Информационная система, решающая задачи оперативного управления предприятием, строится на основе базы данных, в которой фиксируется вся возможная информация о предприятии. Такая информационная система является инструментом для управления бизнесом и обычно называется корпоративной информационной системой.

Информационная система оперативного управления включает в себя массу программных решений по автоматизации бизнес-процессов, имеющих место на конкретном предприятии.

Активное развитие Интернета привело к необходимости создания корпоративных серверов для предоставления различного рода информации о предприятии. Практически каждое уважающее себя предприятие сейчас имеет свой веб-сервер. Веб-сервер предприятия решает ряд задач, из которых можно выделить две основных:

- создание имиджа предприятия;
- максимальная разгрузка справочной службы компании путем предоставления потенциальным и уже существующим абонентам возможности получения необходимой информации о фирме, предлагаемых товарах, услугах и ценах.

Кроме того, использование веб-технологий открывает широкие перспективы для электронной коммерции и обслуживания покупателей через Интернет.

1.4 Общие понятия о базах данных

База данных – информационная модель, которая упорядоченно хранит данные о группе объектов, имеющих одинаковый набор свойств. В базах данных, которые существуют на сегодняшний день, хранятся не только данные, но и информация [2].

Это можно объяснить, если рассмотреть базу данных отдела кадров предприятия, в которой есть все о каждом сотруднике: адреса, должности, данные о зарплате, об образовании, об отпусках. Многие сотрудники организации имеют доступ к базе данных, но среди них нет такого человека, который имеет доступ ко всей базе полностью и может вносить в нее изменения единолично. С помощью методов и средств, имеющихся в базе, сотрудники могут владеть только теми данными, которые входят в его компетенцию. И в результате, сотрудники лишь в пределах своей компетентности, при взаимодействии данных и методов, могут производить ввод и редактирование данных.

С базами данных связанно такое понятие как система управления базой данных (СУБД). Это комплекс программных средств, необходимых для формирования структуры новой базы, наполнения ее содержимым, редактирования содержимого и просмотра информации. Под просмотром информации базы понимается отбор отображаемых данных в соотношении с заданным критерием, их упорядочение, оформление и последующая выдача на устройства вывода или передачи по каналам связи.

Информация, находящаяся в базе данных, упорядочена. Так, в записной книжке все записи хранятся по алфавиту, а в библиотечном каталоге или по алфавиту (алфавитный каталог), или в соответствии с областью знания (предметный каталог).

База, не имеющая никаких данных (пустая база), все равно считается полноценной базой данных. Этот факт имеет методическое значение. В базе нет данных, но информация все же имеется - структура базы, определяющая методы внесения данных и хранение их в базе.

Пример базы данных "некомпьютерной" версии - ежедневник, где каждому календарному дню выделена страница. Если в ней нет записи, он не перестает быть ежедневником, потому что имеет структуру, отличающую его от записной книжки или рабочей тетради.

В базах данных могут содержаться различные объекты. Главными из которых являются таблицы. Самая простая база данных будет состоять из одной таблицы. Соответственно, структура простейшей базы данных тождественно равна структуре ее таблицы.

Структуру двумерной таблицы составляют столбцы и строки. В простейшей базе данных являются поля и записи. Если записей в таблице пока нет, значит, ее структура образована только набором полей. Изменив состав полей базовой таблицы (или их свойства), мы изменяем структуру базы данных и, соответственно, получаем новую базу данных.

Базы данных – это тоже файлы, но работа с ними отличается от работы с файлами других типов, создаваемых прочими приложениями. Мы знаем, что всю работу по обслуживанию файловой структуры берет на себя операционная система. Для базы данных предъявляются особые требования с точки зрения безопасности, поэтому в них реализован другой подход к сохранению данных.

Базы данных – это особые структуры. Информация, которая в них содержится, очень часто имеет общественную ценность. Нередко с одной и той же базой работают тысячи людей по всей стране. От информации, которая содержится в некоторых базах, может зависеть благополучие множества людей. Поэтому целостность содержимого базы не может и не должна зависеть ни от конкретных действий некоего пользователя,

забывшего сохранить файлы перед выключением компьютера, ни от перебоев в электросети.

1.5 Системы управления базами данных

Система программ, разрешающая создавать базу данных, обновлять имеющуюся в ней информацию, дающая удобный доступ к ней с целью просмотра и поиска, называется системой управления базами данных [2].

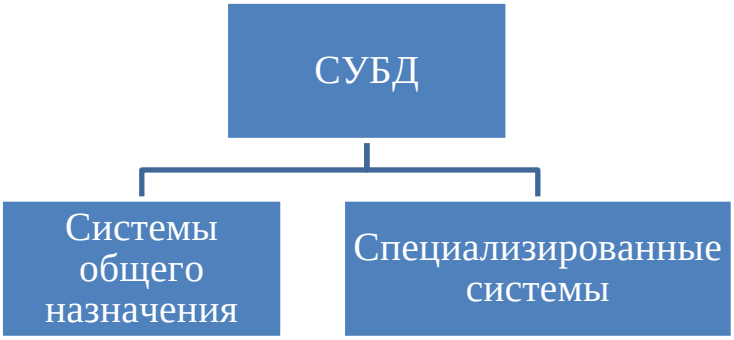

Рисунок 1.4 – Классы систем управления базами данных

Системы общего назначения не относятся к конкретной предметной области или информационным потребностям пользователей, они представляются как программный продукт, работающий на какой-то конкретной модели ЭВМ в определенной операционной системе. Их используют для того, чтобы сократить сроки разработки и сэкономить трудовые ресурсы, за счет предназначения их в качестве инструментального средства создания информационных систем. В процессе осуществления своих функций система управления базами данных непрерывно взаимодействует с базой данных и с другими прикладными, программными продуктами пользователя.

При помощи современных СУБД возможно определить структуру базы и типы данных, также с помощью языка запросов (SQL), можно удалять, вставлять, обновлять и извлекать информацию.

При работе в СУБД возможны следующие режимы: создание, корректировка, поиск, манипулирование. Под манипулированием понимают

такие действия с базой данных, как с целым: просмотр; копирование файлов, например на бумажный носитель; сортировка данных по заданному признаку и т. д.

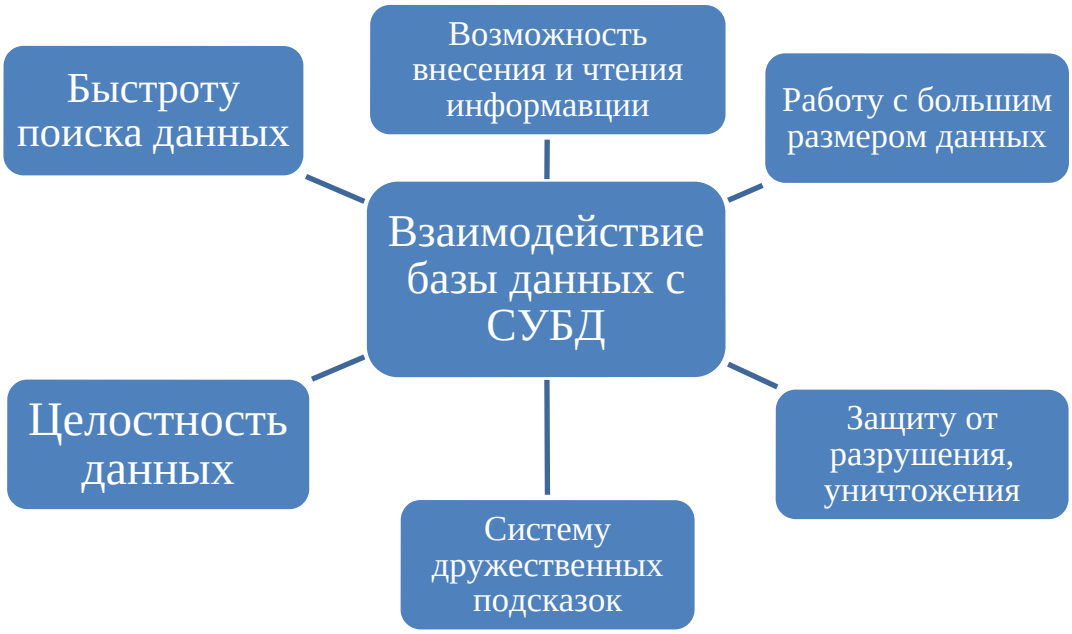

Рисунок 1.5 – Взаимодействие базы данных с СУБД

Обычно с базами данных работают две категории пользователей. Первая категория – проектировщики. Их задача состоит в разработке структуры таблиц базы данных и согласование ее с заказчиком. Кроме таблиц проектировщики разрабатывают и другие объекты базы данных, предназначенные, с одной стороны, для автоматизации работы с базой, а с другой стороны – для ограничения функциональных возможностей работы с базой (если это необходимо из соображений безопасности). Проектировщики не наполняют базу конкретными данными, (заказчик может считать их конфиденциальными и не предоставлять посторонним лицам). Исключение составляет экспериментальное наполнение модельными данными на этапе отладки объектов базы.

Вторая категория исполнителей, работающих с базами данных, - пользователи. Они получают исходную базу данных от проектировщиков и занимаются ее наполнением и обслуживанием. В общем случае пользователи

не имеют средств доступа к управлению структурой базы – только к данным, да и то не ко всем, а к тем, работа с которыми предусмотрена на конкретном рабочем месте.

Соответственно СУБД имеет два режима работы: проектировочный и пользовательский. Первый режим предназначен для создания или изменения структуры базы и создание ее объектов. Во втором режиме происходит использование ранее подготовленных объектов для наполнения базы или получения данных из нее.

На рынке программного обеспечения выделяют такие самые востребованные СУБД: Microsoft Access, Microsoft Visual FoxPro, Orac1e, MySQL.

Приведем некоторую статистику среди популярных СУБД (Рис. 1.6).

В ходе исследования, было выявлено, что лидирующие позиции занимают СУБД с открытым исходным кодом MySQL и Oracle Database.

MySQL — свободная реляционная система управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle. Помимо этого, разработчики создают функциональность по заказу лицензионных пользователей. Именно благодаря такому заказу почти в самых ранних версиях появился механизм репликации. MySQL является решением для малых и средних приложений. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

Oracle Database - это объектно-реляционная система, поддерживающая некоторые технологии, реализующие объектно-ориентированный подход, то есть обеспечивающих управление создания и использования баз данных.

Также не менее популярны такие СУБД, как: MS SQL Server, PostgreSQL. И давно известный для нас Microsoft Access также вошел в десятку используемых СУБД.

Microsoft SQL Server - система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов - Transact-SQL, создан совместно Microsoft и Sybase.

PostgreSQL - свободная объектно-реляционная система управления базами данных. Существует в реализациях для множества UNIX-подобных платформ, включая AIX, различные BSD-системы, HP-UX, IRIX, Linux, macOS, Solaris/OpenSolaris, Tru64, QNX, а также для Microsoft Windows.

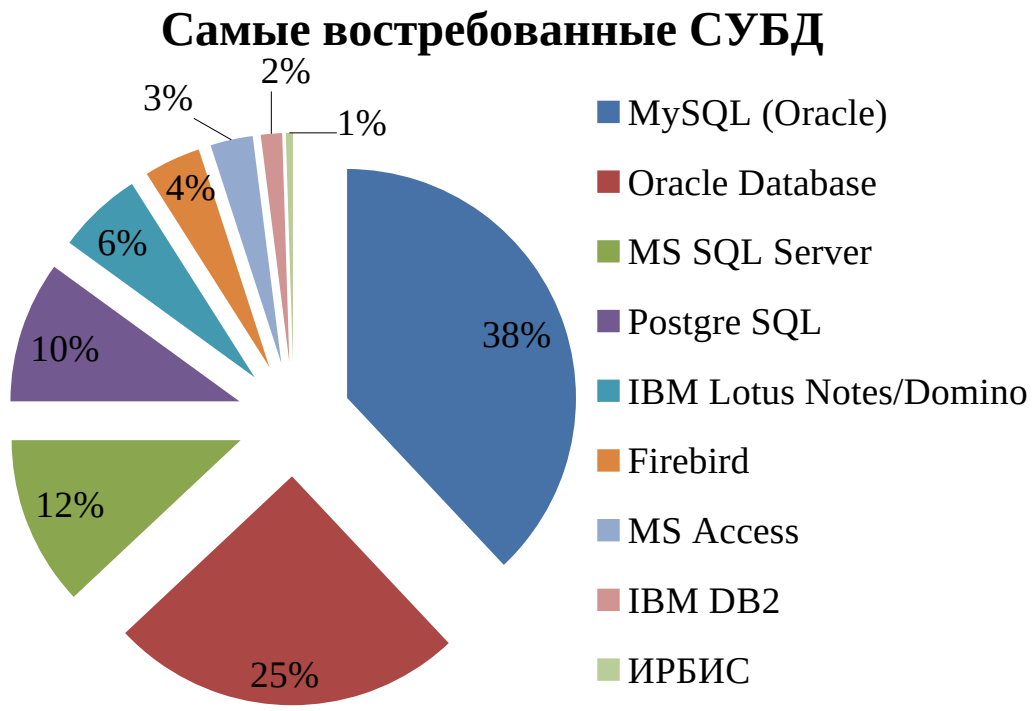

Рисунок 1.6 – Диаграмма востребованных СУБД

1.6 Классификация систем управления базами данных

Возникновение компьютерной техники ознаменовало информационный переворот во всех сферах человеческой деятельности. База данных, где материалы сортируются, систематизируются, что в итоге дает возможность с легкостью найти и предать дальнейшей обработке, была придумана с целью группировки имеющейся информации, которая могла стать ненужным мусором в сети Интернет.

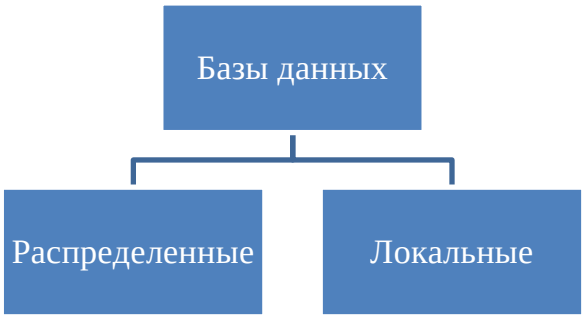

Рисунок 1.7 – Деление СУБД по степени распределенности

Распределенная СУБД содержит несколько пересекающихся или повторяющих друг друга частей, хранящихся на различных ЭВМ вычислительной сети. Локальная же система находится в памяти одной вычислительной системы, и если та является компонентом сети ЭВМ, то возможен распределенный доступ к такой базе. Применяется часто в локальных сетях ПК.

Системы централизованных баз данных с сетевым доступом предполагают различные архитектуры таких систем:

- файл-сервер.

При данной архитектуре выделяется одна из машин сети, которая выступает в роли сервера. В ней хранится централизованная база данных, которая может использоваться совместно остальными рабочими станциями. С их помощью осуществляется доступ пользовательской системы к централизованной базе данных [1].

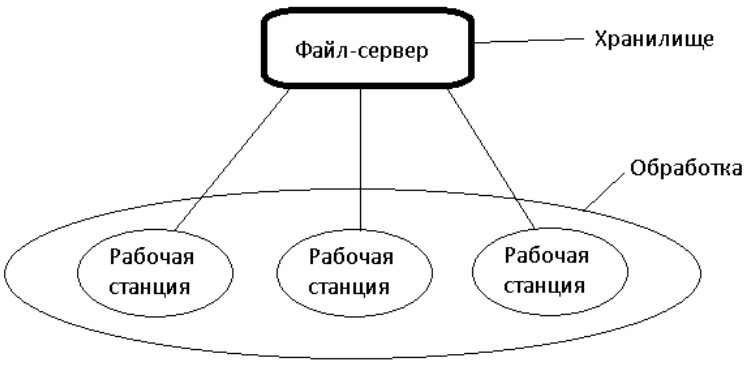

Рисунок 1.8 – Схема архитектуры файл-сервер

Файлы базы данных по запросам пользователей удаленных рабочих станций передаются по сети, где в основном и производится обработка. При большом использовании работы с доступом к одним и тем же данным, производительность такой информационной системы падает.

Пользователь может формировать так же локальные базы данных на рабочих столах, которыми пользуются монопольно. Схема обработки информации по принципу файл-сервер изображена на рисунке 1.8;

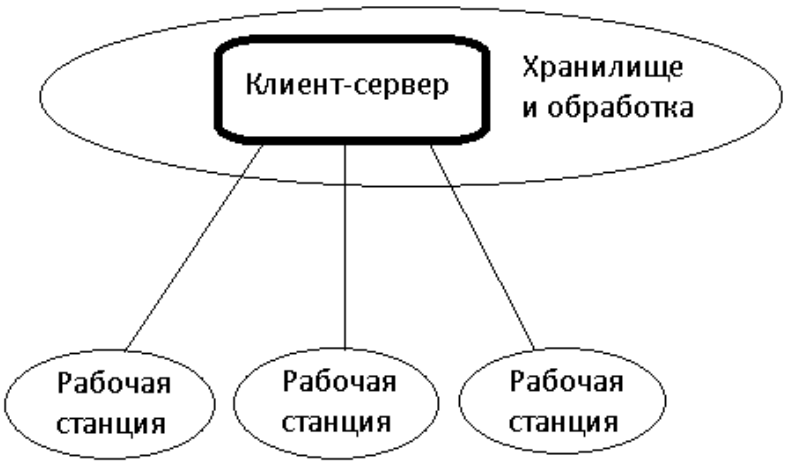

Рисунок 1.9 – Схема архитектуры клиент-сервер

-клиент-сервер (Рис. 1.9). В данной архитектуре центральная машина (сервер) не только хранит централизованную базу данных, но и обеспечивает выполнение основного объема обработки данных. Запросы создаваемые клиентом (рабочей станцией), приводят к поиску, извлечению и отправлению данных с сервере. Они транспортируются по сети от сервера к клиенту. В этой архитектуре чаще всего используется язык запросов SQL.

Связанные между собой группы элементов данных называются записью. Известны три главных типа организации данных и связей между ними: иерархический (в виде дерева), сетевой (в виде графов) и реляционный (в виде таблиц) [2].

В иерархической модели (Рис. 1.10) существует упорядоченность элементов в записи, один элемент считается основным, остальные —

подчиненными. Данные в записи представлены последовательно, как ступеньки лестницы, и поиск данных может осуществляться лишь последовательным "спуском" со ступеньки на ступеньку, что дает процесс поиска нужного элемента трудоемким из-за необходимости последовательно проходить несколько предшествующих иерархических уровней.

Иерархическую модель составляет каталог файлов, находившийся на диске. К такой же базе данных относится родовое генеалогическое дерево.

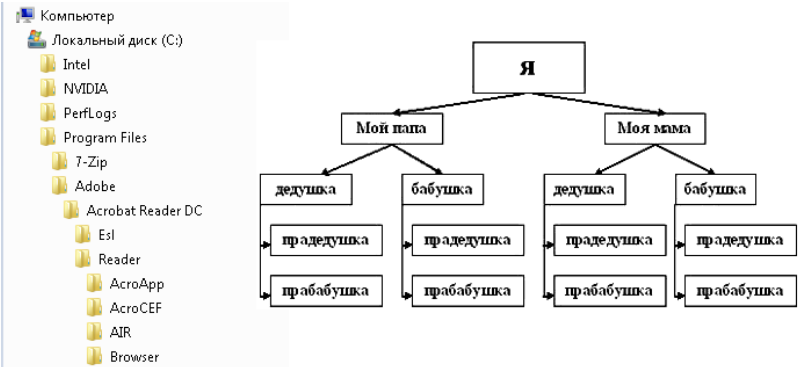

Рисунок 1.10 – Иерархическая модель базы данных

Другая модель базы данных отличающаяся от прежней называется сетевой. Она дает возможность к вертикальным иерархическим связям дополнительно устанавливать горизонтальные связи (Рис. 1.11). Здесь не требуется прохождение всех предшествующих ступеней, что заметно облегчает процесс поиска нужных элементов данных.

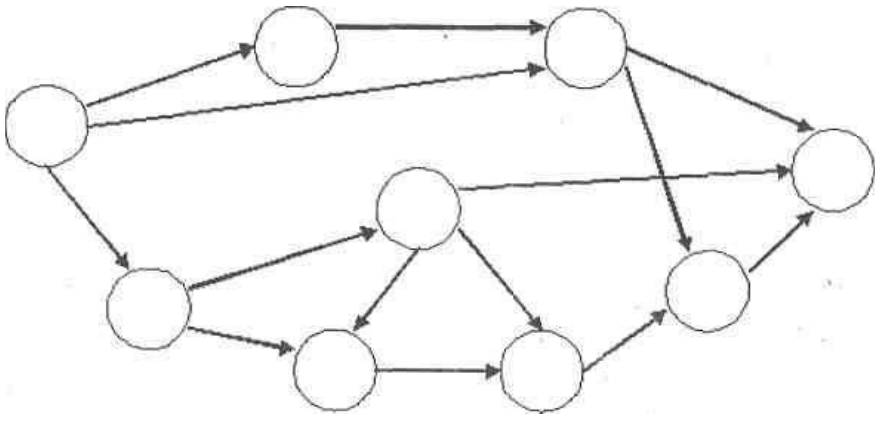

Рисунок 1.11 – Сетевая модель базы данных

Еще одна из моделей организации данных – реляционная, к ней можно отнести и иерархическую, и сетевую.

В реляционной модели (Рис. 1.12) строка прямоугольной таблицы является записью, элементы которой образуют столбцы этой таблицы (поля). В таблице отсутствуют одинаковые строки. Каждый элемент в столбце имеет одинаковый тип, а сам столбец – разные имена.

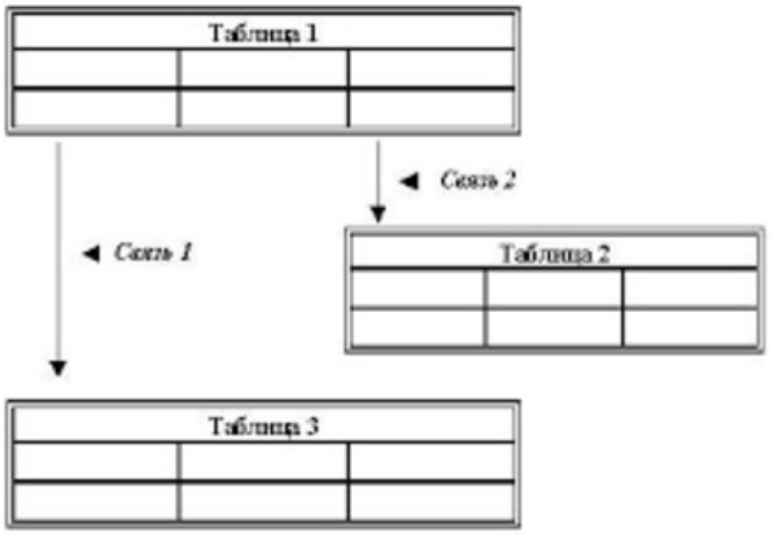

Рисунок 1.12 – Реляционная модель базы данных

Преимущество таких БД – наглядность и понятность организации данных, быстрота поиска необходимой информации. Пример реляционной базы данных – таблица на странице классного журнала, в которой записью является строка с данными о конкретном учащемся, а имена полей (столбцов) указывают, какие данные о каждом учащемся должны быть записаны в ячейках таблицы.

Объектно-ориентированные управляют базами данных, в которых данные моделируются в виде объектов, их атрибутов, методов и классов.

Этот вид СУБД позволяет работать с объектами баз данных так же, как с объектами в программировании в объектно-ориентированных языках программирования. ООСУБД расширяет языки программирования, прозрачно вводя долговременные данные, управление параллелизмом,

восстановление данных, ассоциированные запросы и другие возможности (GemStone).

Объектно-реляционные позволяют через расширенные структуры баз данных и язык запросов использовать возможности объектно-ориентированного подхода: объекты, классы и наследование.

Зачастую все те СУБД, которые называются реляционными, являются, по факту, объектно-реляционными (PostgreSQL, DB2, Oracle, Microsoft SQL Server).

1.7 Структурные элементы базы данных

Концептуальная модель БД описывает сущности, их свойства и связи между ними (не зависит от конкретной СУБД).

Сущность – таблица. В базе может быть как одна таблица, так и набор из таблиц, которые характеризуют описанные объекты, благодаря находящимся в них данным.

Таблица 1.1 – Таблица «Сотрудники»

Код	Фамилия	Имя	Отчество	Дата рождения	Адрес
111111	Митрошин	Митрофан	Никитович	22.06.1978	40 лет Победы,27
123123	Васечкин	Николай	Петрович	30.12.1970	Уральская,30
123456	Прохоров	Сергей	Михайлович	12.05.1969	Уральская,27
123789	Сатарова	Светлана	Васильевна	08.10.1971	Московская,20
321123	Мартюшев	Василий	Евгеньевич	22.05.1979	Уральская,39-45
456123	Воронин	Константин	Сергеевич	04.10.1985	Удмуртская,4

В них имеется закрепленное количество полей и меняющееся число записей (Рис.1.13). Сущность также называют отношением.

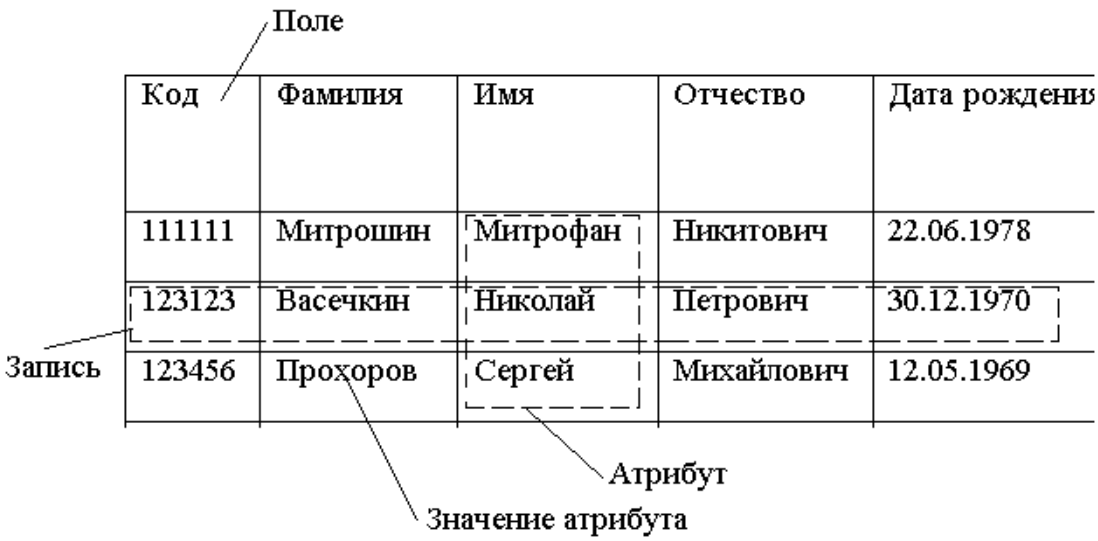

Рисунок 1.13 – Обозначения реляционной таблицы

Как видно из таблицы 1.1, она обладает следующими свойствами:

- строка таблицы – один элемент данных (сведения об одном сотруднике);

- все столбцы - однородные, т.е. имеют одинаковую длину и тип (в столбце Имя отображаются имена сотрудников символьного типа);

- каждый столбец имеет свое имя;

- повторение строк не допускаются;

- порядок следования строк и столбцов в таблице может быть произвольным.

В любой реляционной базе данных у таблицы должен быть первичный ключ. Он должен быть уникальным и однозначно определять запись. По значению ключа можно отыскать единственную запись. Ключи нужны также для упорядочивания информации в БД.

Модели реляционных баз данных имеют только один потенциальный ключ, это и будет primary key. Представляющий собой первичный ключ – это столбец сущности или набор атрибутов, с помощью которого можно получить доступ к данным конкретной строки. Он должен быть единственным и в полях не могут быть пустые значения. Если содержится

один атрибут, то он считается простым, в ином случае будет составляющим. Существует и внешний ключ (foreign key).



Рисунок 1.14 – Первичный и внешний ключи

Таблицы реляционной базы данных должны отвечать требованиям нормализации отношений. Нормализация отношений – формальный аппарат ограничений на формирование таблиц, который позволяет устранить дублирование, обеспечивает непротиворечивость хранимых в базе данных, уменьшает трудозатраты на ведение базы данных [2].

Первая нормальная форма (Рис.1.15). Таблица находится в этой форме, когда в любом поле ни одна из записей не содержит более одного значения, а ключевые поля не содержат пустых значений.



Рисунок 1.15 – Первая нормальная форма

Вторая нормальная форма. Таблица находится в этой форме, когда удовлетворяются требования первой и поля не входящие в первичный ключ связаны полной функциональной зависимостью с первичным ключом. Чтобы таблица получила вид этой формы, нужно определить функциональную зависимость полей. Функциональная зависимость полей - это зависимость, при которой в экземпляре информационного объекта определенному значению ключевого реквизита соответствует только одно значение описательного реквизита.

В третьей нормальной форме (Рис. 1.16) таблица удовлетворяет второй нормальной форме, поля не зависят от ее ключевых полей.

Код сотрудника	Фамилия	Служба	Начальник
111111	Митрошин	Бухгалтерия	Полыгаева
123123	Васечкин	Гараж	Никоноров
123456	Прохоров	Отдел кадров	Манилов
123789	Сатарова	Бухгалтерия	Полыгаева
321123	Мартюшев	Отдел кадров	Манилов
456123	Воронин	Гараж	Никоноров

=

Код сотрудника	Фамилия	Служба
111111	Митрошин	Бухгалтерия
123123	Васечкин	Гараж
123456	Прохоров	Отдел кадров
123789	Сатарова	Бухгалтерия
321123	Мартюшев	Отдел кадров
456123	Воронин	Гараж

+

Служба	Начальник
Бухгалтерия	Полыгаева
Гараж	Никоноров
Отдел кадров	Манилов

Рисунок 1.16 – Третья нормальная форма

Реляционные таблицы связываются друг с другом, последовательно, информация может извлекаться одновременно из нескольких таблиц. Это нужно, чтобы уменьшился объем базы данных.

При наличии одинаковых столбцов обеспечивается связь каждой пары таблиц.

Существуют следующие типы информационных связей:

- один-к-одному, когда один атрибут первой таблице, соответствует одному элементу другой, и наоборот;

- один-ко-многим, показывает, что атрибут одной таблицы соответствует с несколькими атрибутам другой.

- многие-ко-многим, предполагает, что одному атрибуту первой таблицы соответствует несколько атрибутов второй таблицы и наоборот.

Чтобы создать свою базу данных, нужно использовать один из инструментов моделирования, решить, на какую информацию будет ориентирована база данных, спроектировать таблицы и реляционные одно- и множественные связи между данными, заполнить ячейки сущностей и установить первичный и внешние ключи.

1.8 Обзор продуктов для разработки и администрирования баз данных

Многие компании создают различные многофункциональные приложения для облегчения управления, разработки и администрирования баз данных.

Большинство реляционных баз данных, за исключением MS Access, состоят из двух отдельных компонентов: «back-end», где хранятся данные и «front-end» — пользовательский интерфейс для взаимодействия с данными.

В Интернете можно найти много продуктов для разработки и администрирования баз данных MySQL. Мы рассмотрим 10 самых популярных инструментов.

Самыми популярными являются такие программы (Рис. 1.17), как PHPMyAdmin, dbForge Studio for MySQL и HeidiSQL, так как они имеют в наличии достаточно простой функционал по моделированию баз, ну и являются бесплатными. Такие программы, как Navicat и SQLyog, являются платными и поэтому менее востребованы, хотя не многим уступают лидерам. Далее приведем краткий обзор программ.

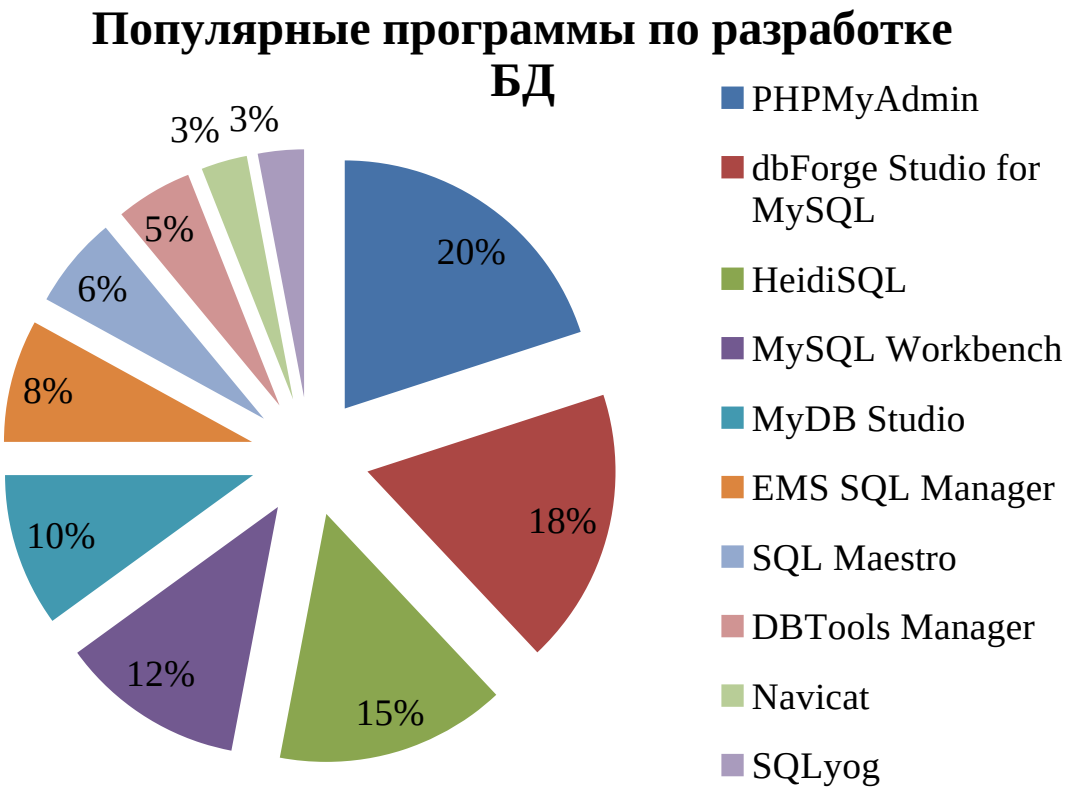

Рисунок 1.17 – Программы по разработке баз данных

1. PHPMyAdmin — бесплатное приложение с открытым кодом, предназначенное для администрирования СУБД MySQL. PHPMyAdmin представляет собой веб-интерфейс с помощью которого можно администрировать сервер MySQL, запускать команды и просматривать содержимое таблиц и БД через браузер. PHPMyAdmin дает возможность управлять СУБД MySQL без непосредственного ввода SQL команд и интегрировать PHPMyAdmin в собственные разработки благодаря лицензии GNU General Public License и другие возможности.

2. dbForge Studio for MySQL — инструмент, представляющий интерес как для пользователей MySQL, так и для разработчиков БД. С его помощью можно легко автоматизировать рутинную работу и сэкономить время. Сегодня dbForge Studio for MySQL представлен в трех редакциях: Express, Standard и Professional, что позволяет выбрать тот инструмент, который нужен.

3. HeidiSQL — бесплатный инструмент для управления базами данных. Достойная альтернатива PHPMyAdmin, которая позволяет создавать и редактировать таблицы, представления, триггеры, процедура, а также просматривать и редактировать данные. Также HeidiSQL предоставляет возможность экспорта данных как в SQL файл, так и в буфер обмена на других серверах. С помощью HeidiSQL можно подключаться к серверу через командную строку, а также есть возможность пакетной оптимизации и восстановления таблиц.

4. MySQL Workbench (разработка компании Sun Systems/Oracle) распространяется под свободной лицензией — Community Edition и с ежегодной оплачиваемой подпиской — Standard Edition. Последняя включает в себя дополнительные возможности, которые способны существенно улучшить производительность, как разработчиков, так и администраторов баз данных. В Workbench представлено наличие простого и функционального механизма по созданию связей между полями таблиц, среди которых реализована связь «многие-ко-многим».

5. MyDB Studio — бесплатный инструмент для администрирования БД MySQL, который позволяет создавать, редактировать и удалять записи, таблицы и базы данных. Работает исключительно на платформе Windows. В MyDB Studio возможно подключаться к неограниченному количеству баз данных и подключение по SSH каналам.

1.9 Нечеткие базы данных и запросы в них

Большая часть данных, обрабатываемых в современных информационных системах, носят четкий, числовой характер. Однако в запросах к реляционным базам данных, которые пытается формулировать человек, часто присутствуют неточности и неопределенности. Поэтому нечеткие запросы к базам данных – это перспективное направление в современных системах обработки информации.

Понятия «нечеткость» и «неопределенность» имеют разный смысл. Неопределенность возникает из-за неполноты знаний или недостатка информации, относящихся к некоторому событию или наличию свойства какого-либо объекта. Нечеткость относится к способу описания самого события, свойства.

Современные СУБД позволяют работать только с четкими базами данных и четкими запросами. В настоящее время существует проблема разработки математического описания, методов и средств обработки данных в условиях нечеткости и неопределенности.

Применение и внедрение теории нечетких систем в задачах поиска информации переживает в настоящее время период бурного роста.

Вопросы плохо формализуемой математической корректности описания, слабо структурированной информации, с последующей обработкой всегда находились в поле зрения специалистов по нечетким системам. А на начальном этапе своего развития теория нечетких множеств представлялась как средство описания такой информации. Неслучайно первой «нечеткой» теорией была теория лингвистической переменной Заде. Поэтому вопросом в рамках обработки нечеткой информации явился вопрос поиска информации в базах данных, описания объектов которых есть лингвистические описания объектов предметной области.

Проблемы поиска информации в нечетких базах данных долгое время оставалась на периферии проводимых исследований. И только в начале 90-х годов они вновь привлекли значительный интерес, как теоретиков, так и прикладников.

Нечеткие запросы – перспективное направление в современных системах обработки информации, они появились в связи с необходимостью смягчить булеву логику в реляционных базах данных. Данный инструмент дает возможность формулировать запросы на естественном языке, например, «Вывести список недорогих предложений о съеме жилья близко к центру

города», что невозможно при использовании стандартного механизма запросов.

Нечеткие запросы целесообразно использовать в областях, где осуществляется выбор информации из базы данных с использованием качественных критериев и нечетко сформулированных условий. Например, в прямом маркетинге, при выборе туристических услуг, подборе объектов недвижимости.

Механизм нечетких запросов к реляционным базам данных базируется на теории нечетких множеств, которая была впервые предложена Лотфи Заде [3].

Четкая база данных – совокупность записей, значения атрибутов которых есть либо строковые, либо численные значения, однозначно понимаемые пользователями.

Под четким запросом понимается логическое высказывание, термы которого выражаются обычными средствами теории нечетких множеств и двузначной четкой математической логикой. Это означает, что можно либо перечислить значения признаков интересующих нас объектов, либо указать границы изменения параметров признаков и связать данные пары «признак – значение» логическими связками.

Нечеткий запрос, в отличии от четкого, может содержать термы с нечеткими значениями. Например, значением признака «Размер» могут быть «Большой», «Не большой и не маленький»; значением признака «Возраст» – «Молодой» , «Более или менее молодой» и т.п.

Подобным образом четкие базы данных отличаются от нечетких: атрибуты последних могут иметь нечеткие значения.

Продемонстрируем ограниченность четких запросов на следующем примере.

Пусть требуется получить сведения о квартире общей площадью 36кв.м. и стоимостью не выше 1300т.р.

Данный запрос можно записать на языке SQL следующим образом: select *from Flat where(Square=36 and Cost <=1300)

Квартира площадью 37 кв.м. и стоимостью 1300 т.р. не попадет в результат запроса, хотя ее характеристики почти удовлетворяют требованиям запроса. Нечеткие запросы помогают справиться с подобными проблемами «пропадания» информации.

Для реализации таких запросов понадобится формализовано описать множества значение термов логического выражения. Моделью таких структур является лингвистическая переменная и функция принадлежности.

Лингвистической переменной называется набор

$$< \beta, \mathfrak{X}, T, G, M > \qquad (1.1)$$

где β – имя лингвистической переменной;

$\mathfrak{X}$ – базовое множество;

T – множество ее значений (терм-множество), представляющие имена нечетких переменных, областью определения которых является множество $\mathfrak{X}$

G – синтаксическая процедура, позволяющая оперировать элементами терм-множества T, в частности, генерировать новые термы (значения);

M – семантическая процедура позволяющая преобразовывать новые значения лингвистической переменной, образованной процедурой G, в нечеткую переменную, т.е. сформировать соответствующее нечеткое множество.

Функцией принадлежности называется функция, которая позволяет вычислить степень принадлежности произвольного элемента множества $\mathfrak{X}$ к нечеткому множеству F [3].

Далее появляется проблема определения функции принадлежности. Этот вопрос находится за пределами теории нечетких множеств и больше относится к теории экспертного оценивания и методов обработки экспертной информации.

Рассмотрим базу данных «Агентство недвижимости».

Предположим, что вся информация находится в одной таблице:

Таблица 1.2 – Таблица «Квартиры»

№	Адрес	Кол-во комнат	Общая площадь	Балкон	Цена
1	Лебедевой,54	1	31	0	1300
2	Марковского, 57	1	24	0	950
3	Белинского,3	1	28	0	1380
4	Горького,38	1	28	0	1350
5	Горького,14	1	32	1	1550
6	Копылова,42	1	34	1	1520
7	Кецховели,57	1	32	1	1450
8	Красной армии,10	1	42	1	1750
9	Копылова,78	1	30	1	1630
10	Тотмина,5	1	34	1	1300
11	Кравченко,8	1	42	0	1480

Для реализации нечетких запросов формализуем две лингвистические переменные: «цена» и «площадь».

Переменная «цена» имеет вид:

$$< \beta, \mathfrak{X}, T, G, M > \qquad (1.2)$$

где β – цена;

T={«дешевая», «средняя», «дорогая»};

$\mathfrak{X}$=[800;1800];

G – процедура образования новых термов с помощью связок «и», «или» и модификаторов «не», «очень», «более-менее»;

M – процедура задания на $\mathfrak{X}$ нечетких множеств A=«дешевая», B=«средняя» и C=«дорогая», а также нечетких множеств для термов из G(T) соответственно правилам трансляции нечетких связок и модификаторов и других операций над нечеткими множествами.

Зададим функции принадлежности для переменной «цена» трапециевидного вида:

$$\mu_A(x) = \begin{cases} 1, x \in [800, 950] \\ \frac{1050-x}{150}, x \in (950, 1050] \\ 0, x > 1050 \end{cases} \quad (1.3)$$

$$\mu_B(x) = \begin{cases} 0, x < 1000, x > 1500 \\ \frac{x-1000}{150}, x \in [1000, 1150) \\ 1, x \in [1150, 1350] \\ \frac{1500-x}{150}, x \in (1350, 1550] \end{cases} \quad (1.4)$$

$$\mu_C(x) = \begin{cases} 0, x < 1400] \\ \frac{x-1400}{100}, x \in [1400, 1500) \\ 1, x \in [1500, 1800] \end{cases} \quad (1.5)$$

Графики функций принадлежности приведены на рисунке 1.18.

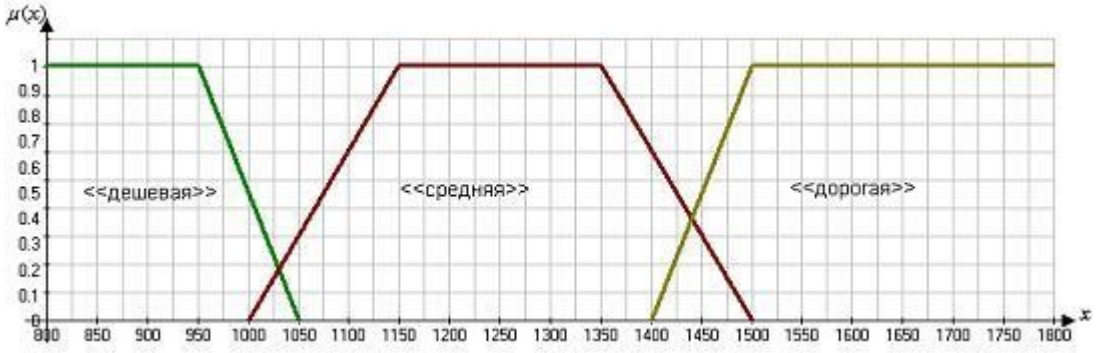

Рисунок 1.18 – Функции принадлежности термов переменной «цена»

Переменная «площадь» имеет вид:

$$< \gamma, \mathcal{Y}, T, G, M >, \quad (1.6)$$

где γ – площадь;

T={«маленькая», «средняя», «большая»};

$\mathcal{Y}$ =[20;50];

G – процедура образования новых термов с помощью связок «и», «или» и модификаторов «не», «очень», «более-менее»;

M – процедура задания на $\mathcal{Y}$ нечетких множеств D=«маленькая», E=«средняя» и F=«большая», а также нечетких множеств для термов из G(T)

соответственно правилам трансляции нечетких связок и модификаторов и других операций над нечеткими множествами.

Зададим функции принадлежности для переменной «площадь» трапециевидного вида:

$$\mu_D(y) = \begin{cases} 1, x \in [2, 24] \\ \frac{30-x}{6}, x \in (24, 30] \\ 0, x > 30 \end{cases} \quad (1.7)$$

$$\mu_E(y) = \begin{cases} 0, x < 29, x > 40 \\ \frac{x-29}{4}, x \in [29, 33) \\ 1, x \in [33, 38] \\ \frac{40-x}{2}, x \in (38, 40] \end{cases} \quad (1.8)$$

$$\mu_F(y) = \begin{cases} 0, x < 39] \\ \frac{x-39}{3}, x \in [39, 42) \\ 1, x \in [42, 50] \end{cases} \quad (1.9)$$

Графики функций принадлежности приведены на рисунке 1.19.



Рисунок 1.19 – Функции принадлежности термов переменной «площадь»

Вычислив степени принадлежности для каждой квартиры, таблицу можно преобразовать к следующему виду:

Таблица 1.3 – Таблица «Квартиры» со степенями принадлежности

№	Адрес	Кол-во комнат	Балкон	Площадь			Цена		
				μ_D	μ_E	μ_F	μ_A	μ_B	μ_C

1	Лебедевой,54	1	0	0	0,5	0	0	1	0

Продолжение табл. 1.3

№	Адрес	Кол-во комнат	Балкон	Площадь			Цена		
				μ_D	μ_E	μ_F	μ_A	μ_B	μ_C
2	Марковского, 57	1	0	1	0	0	1	0	0
3	Белинского,3	1	0	0,33	0	0	0	0,8	0
4	Горького,38	1	0	0,33	0	0	0	1	0
5	Горького,14	1	1	0	0,75	0	0	0	1
6	Копылова,42	1	1	0	1	0	0	0	1
7	Кецховели,57	1	1	0	0,75	0	0	0,33	0,5
8	Красной армии,10	1	1	0	0	1	0	0	1
9	Копылова,78	1	1	0	0,25	0	0	0	1
10	Тотмина,5	1	1	0	1	0	0	1	0
11	Кравченко,8	1	0	0	0	1	0	0,13	0,8

Построим для нее четкие и нечеткие запросы.

Пример 1.

Четкий запрос: «Найти однокомнатные квартиры стоимостью 1350т.р.»

На языке SQL:

select *from Flat where(Room=1 and Cost=1350)

Результат такого запроса будет следующим:

Таблица 1.4 – Результат четкого запроса

№	Адрес	Кол-во комнат	Балкон	Площадь			Цена		
				μ_D	μ_E	μ_F	μ_A	μ_B	μ_C
4	Горького,38	1	0	0,33	0	0	0	1	0

Аналогичный нечеткий запрос: «Найти однокомнатные квартиры среднего ценового диапазона»

На SQL-подобном языке:

select *from Flat where(Room=1 and Cost=«средняя»)

Результат приведен в таблице 1.5.

Таблица 1.5 – Результат нечеткого запроса

№	Адрес	Кол-во комнат	Балкон	Площадь			Цена		
				μ_D	μ_E	μ_F	μ_A	μ_B	μ_C
1	Лебедевой,54	1	0	0	0,5	0	0	1	0
4	Горького,38	1	0	0,33	0	0	0	1	0
10	Тотмина,5	1	1	0	1	0	0	1	0
3	Белинского,3	1	0	0,33	0	0	0	0,8	0
7	Кецховели,57	1	1	0	0,75	0	0	0,33	0,5
11	Кравченко,8	1	0	0	0	1	0	0,13	0,8

Пример 2.

Четкий запрос: «Найти однокомнатные квартиры с балконом, площадью 36кв.м. и стоимостью 1350т.р»>

На языке SQL:

select *from Flat where(Room=1 and Balcony=1 and Square=36 and Cost=1350)

Записей, удовлетворяющих этим условиям не нашлось.

Аналогичный нечеткий запрос: «Найти однокомнатные квартиры средней площадью с балконом и средним ценовым диапазоном»

На SQL-подобном языке:

select *from Flat where(Room=1 and Balkony=1 and Square=«средняя» and cost=«средняя»)

Т.к. два параметра запроса определены степенью принадлежности, то оператор and определяется как декартово произведение нечетких множеств.

Декартово произведение $A_1 \times A_2 \times \ldots \times A_n$ нечетких множеств A_i в $\mathfrak{X}_i, i = 1, \ldots, n$ определяется как нечеткое множество A в декартовом произведении $\mathfrak{X} = \mathfrak{X}_1 \times \cdots \times \mathfrak{X}_n$ с функцией принадлежности вида:

$$\mu_A(x) = min\{\mu_{A_1}(x_1), \mu_{A_2}(x_2), \ldots, \mu_{A_n}(x_n)\}, x = (x_1, \ldots, x_n) \in \mathfrak{X}$$

Таким образом, получим следующий результат:

Таблица 1.6 – Результат нечеткого запроса

№	Адрес	Кол-во комнат	Балкон	Площадь			Цена		
				μ_D	μ_E	μ_F	μ_A	μ_B	μ_C
7	Кецховели,57	1	1	0	0,75	0	0	0,33	0,5
10	Тотмина,5	1	1	0	1	0	0	1	0

Аналогичным образом нечеткие запросы можно использовать при выборе туристических услуг или подборе объектов недвижимости.

Инструмент нечетких запросов позволяет согласовать формальные критерии и неформальные требования к кругу потенциальных клиентов и задавать интервалы выбора потенциальных клиентов как нечеткие множества. В таком случае клиенты, не удовлетворяющие какому-то одному критерию, могут быть выбраны из базы данных, если они имеют хорошие показатели по другим критериям.

Можно сделать следующие выводы:

- нечеткие запросы позволяют расширить область поиска в соответствии с изначально заданными человеком ограничениями;
- недостатком нечетких запросов является относительная субъективность функций принадлежности.

2 ПРИМЕНЕНИЕ НЕЧЕТКИХ БАЗ ДАННЫХ В ИНФОРМАЦИОННЫХ СИСТЕМАХ

2.1 Нечеткая логика и нечеткие системы

При решении большинства задач, с которыми многие люди сталкиваются повседневно, возникает необходимость приближенного задания условий и, соответственно, получаются столь же приближенные ответы. В самых разных сферах деятельности: управление инвестиционными портфелями, планирование финансовой деятельности предприятия, оптимизация товарооборота, оптимизация финансовых потоков, оптимизация информационных потоков, оценка эффективности рекламной компании, оценка влияния политических и социальных событий на поведение рынка, а также многих, многих прочих задачах требуются такие инструменты, которые могут подобные вычисления проводить, причем достаточно быстро и с приемлемой точностью.

Речь далее пойдет об инструментах, использующих принципы нечеткой логики (fuzzy logic). Встречается также другой термин – нечеткое представление(fuzzy thinking).

Управляя автомобилем, двигаясь в плотном городском потоке, мы заняты разгоном, торможением, маневрированием, соблюдением правил движения и т.д. В этот момент, на такие вопросы, как температура тормозов или давление масла в гидроусилителе, вряд ли можно ответить что-то определенное. Все агрегаты автомобиля интересуют нас постольку, поскольку они способствуют этому процессу, так как просто нет ни времени, ни возможности отвлекаться на детали [4].

Иными словами, рассматривая поездку с точки зрения системы, можно сказать, что в большей степени волнуют принципы действия этой системы, то есть насколько способен автомобиль доставить из точки А в точку Б за заданное время и при заданных известных условиях. И в меньшей степени

волнует конструкция этой системы, то есть то, каким способом автомобиль решает эту задачу.

При более детальном изучении вопроса можно заметить, что в подавляющем большинстве задач входные условия и критерии оценок непрерывно изменяются.

Если вернуться к примеру автомобиля, то когда мы прикидываем расход топлива при езде по городу (целевая функция), мы строим утверждения примерно так:

«обычно я двигаюсь по городу из точки А в точку Б со скоростью 60 км/час», подразумевая при этом, что в некоторых местах скорость выше, а в других – близка к нулевой;

«при этом обороты двигателя примерно 2500 об/мин», понимая, что на светофорах это холостой ход, а при разгоне – близки к максимальным;

В результате, замеряя средний расход топлива в литрах на 100 км, можно обнаружить, что он существенно отличается от паспортных значений. Это отличие не устраивает, но есть варианты решения этой проблемы

Ответ с точностью до миллилитра не нужен, важно просто минимизировать целевую функцию. Иными словами можно определить некоторую степень точности решения задачи, при соблюдении которой устроит результат.

Если задаться целью свести значение целевой функции (расхода топлива) к минимуму, не прибегая к созданию системы, по цене сравнимой со стоимостью автомобиля, понадобится инструмент, который может решать такую задачу при непрерывно меняющихся входных значениях и который может легко подстраиваться под изменение оценочных критериев (например, движение в городе/на трассе). Причем инструмент должен быть прост в управлении и понятен без долгого изучения специальных дисциплин.

Прежде, чем приступить к описанию конкретных пакетов, необходимо иметь в виду следующее обстоятельство: пути решения каждой конкретной задачи могут быть весьма разнообразны. Современный математический

аппарат предоставляет целый спектр методов, приемов и инструментов для решения практически любой задачи. Все они воплощены в виде алгоритмов в разнообразных программных продуктах.

Коротко перечислим отличительные преимущества fuzzy-систем по сравнению с прочими:

- возможность оперировать входными данными, заданными нечетко: например, непрерывно изменяющиеся во времени значения (динамические задачи), значения, которые невозможно задать однозначно (результаты статистических опросов, рекламные компании и т.д.);

- возможность нечеткой формализации критериев оценки и сравнения: оперирование критериями «большинство», «возможно», «предпочтительно» и т.д.;

- возможность проведения качественных оценок как входных данных, так и выводимых результатов: вы оперируете не только собственно значениями данных, но и их степенью достоверности и ее распределением;

- возможность проведения быстрого моделирования сложных динамических систем и их сравнительный анализ с заданной степенью точности: оперируя принципами поведения системы, описанными fuzzy-методами, вы во-первых, не тратите много времени на выяснение точных значений переменных и составление уравнений, которые их описывают, во-вторых, можете оценить разные варианты выходных значений [5].

Математический аппарат, предоставляющий такие возможности, детально описан в специальной литературе как комбинация множественных и вероятностных приемов.

В программных пакетах и системах этот аппарат реализован в полной мере, но внешне не виден, спрятан "за кадром", что делает процесс освоения этих инструментов более доступным и интуитивно понятным.

2.2 Электронная таблица Fuzi Calc

Пакет Fuzi Calc фирмы FuziWare сравнительно молод (1995 год), но успел завоевать популярность как недорогой инструмент, позволяющий проводить быстрые (прикидочные) расчеты в различных областях бизнеса и получать результаты с вполне приемлемой степенью точности [6].

Рабочая среда пакета выглядит так:

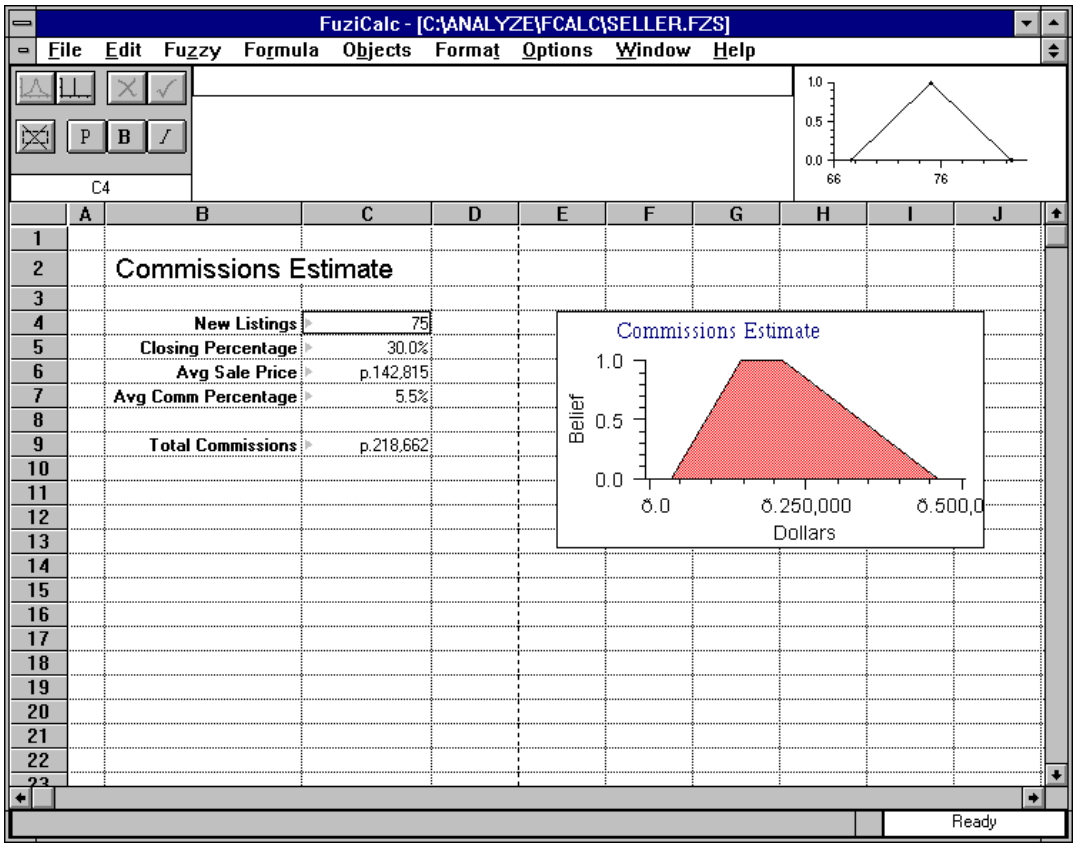

Рисунок 2.1 – Рабочая среда Fuzi Calc

На рисунке (Рис. 2.1) показан пример решения задачи оценки прибыли при работе фирмы на рынке недвижимости.

Постановка задачи такова: менеджер планирует деятельность фирмы, работающей на рынке недвижимости на следующий год. Целевая функция задачи – диапазон прибыли, на который можно рассчитывать.

Для решения задачи существуют четыре нечетких утверждения, выявленных из статистики деятельности фирмы за несколько прошедших лет:

1. В течение каждого года в фирму обращаются примерно 75 потенциальных клиентов.

2. Из потенциальных клиентов примерно 30 % совершают сделки.

3. Стоимость недвижимости, фигурирующей в сделках, составляет примерно $142815.

4. За проведенную сделку с каждого клиента отчисляется примерно 5.5 % комиссионных.

Если просто перемножить эти четыре значения, мы получим цифру $176733. Такова сумма прибыли за год, если ее вычислять обычным способом.

При этом менеджер ясно осознает, когда он говорит о 75 потенциальных клиентах в год, то в разные года получается по-разному.

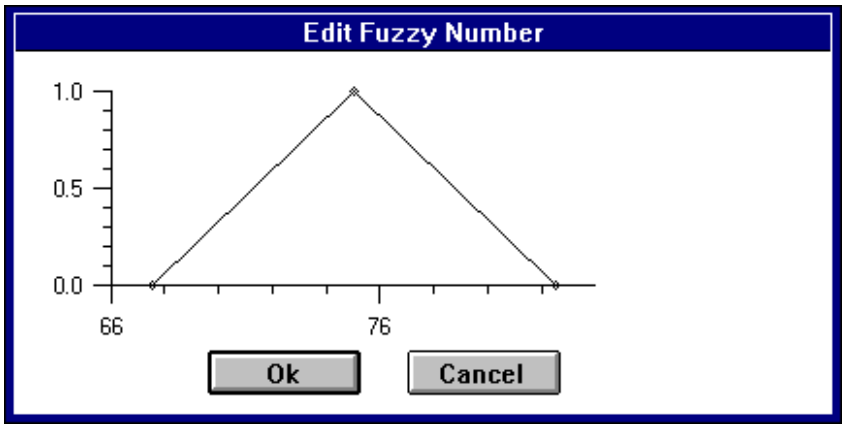

Рисунок 2.2 – График распределения потенциальных клиентов

Но совершенно точно известно, что никогда не бывало меньше 67 и никогда не было больше 87. (На графике 2.2 по вертикали - достоверность значения, отложенного по горизонтали).

Когда он говорит о 30 % потенциальных клиентов, совершающих сделки, на самом деле это выглядит так:

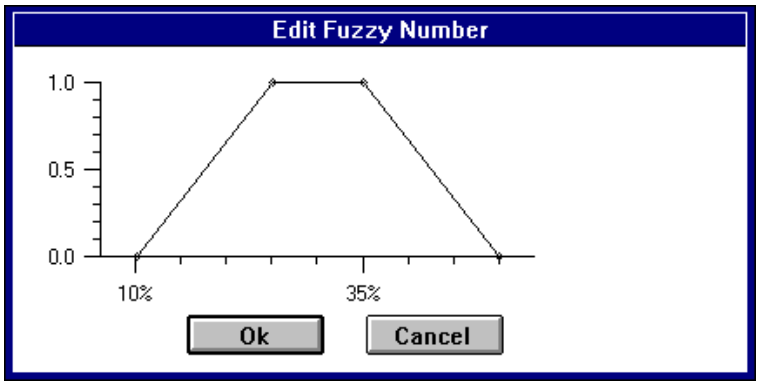

Рисунок 2.2 – График распределения потенциальных клиентов, совершающих сделки

Соответственно это значение изменяется в пределах 25-35%, но никогда не было меньше 10% и никогда не было больше 50 %.

Когда он говорит о стоимости недвижимости в $142815, то фактически стоимость различных объектов сделок распределяется так:

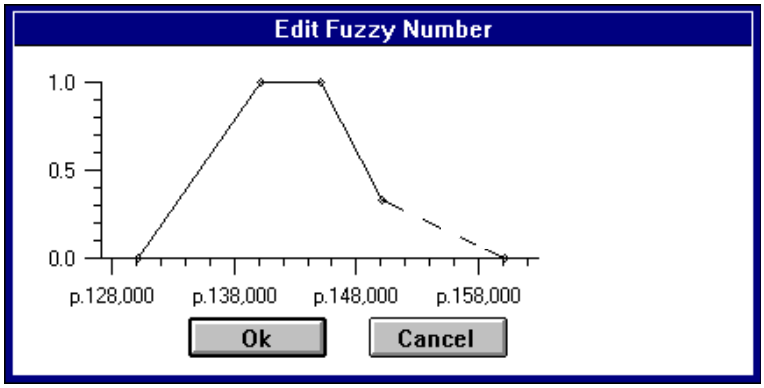

Рисунок 2.3 – График распределения стоимости объектов сделок

Поэтому чаще всего происходят сделки по цене $138000 – $140000 , но никогда не дешевле $130000 и не дороже $150000. Обратите внимание на излом графика в зоне, обозначенной пунктирной линией и уход в точку, соответствующую $160000. Здесь менеджер подразумевает, что в следующем году фирма попробует проводить сделки с более дорогой недвижимостью, но не свыше $160000. В противном случае точки перелома не было бы вообще, а график упирался бы в точку, соответствующую цене $150000.

Когда он говорит о комиссионных в 5.5 %, то на самом деле в разных случаях размер комиссионных распределяется так:

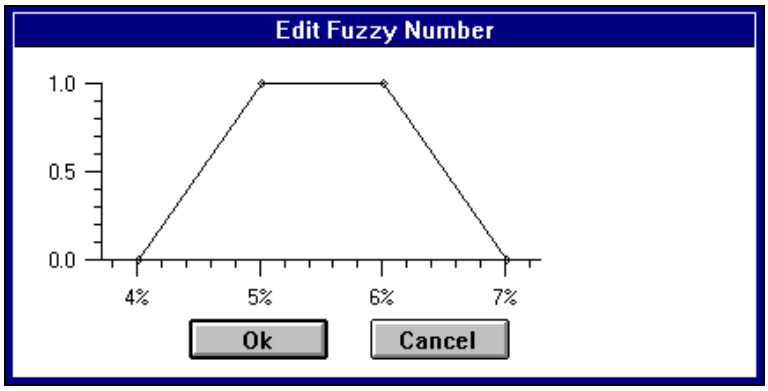

Рисунок 2.4 – График распределения комиссионных

То есть чаще всего их размер лежит в пределах 5-6% , но не меньше 4% и не больше 7% .

Система при такой постановке задачи произведение значений вычисляет перемножением интегралов, описывающих данные кривые. Поскольку типы кривых известны, число их не бесконечно, и применение 5 стандартных функций, описывающих эти кривые делает вычисления практически мгновенными. Самое главное в том, что изменяя входные значения, Вы не меняете способ их вычисления (системы уравнений, описывающих функции).

В данном примере Вы видите, что, построив 4 нечетких утверждения и перемножив их значения (определение прибыли), ответ можно увидеть так:

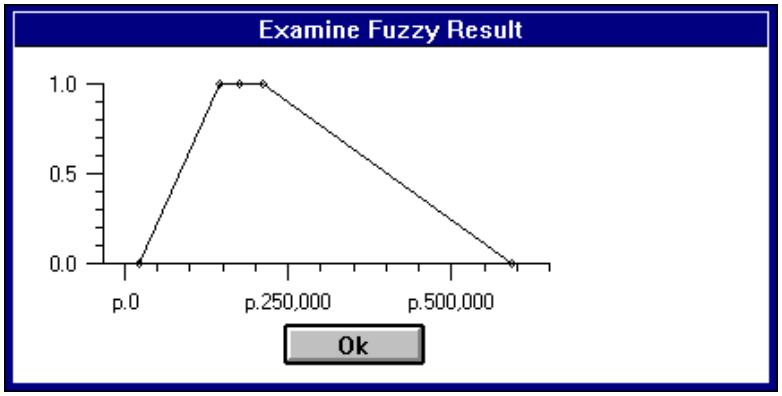

Рисунок 2.5 – График распределения прибыли

То есть наиболее достоверное значение прибыли будет находиться в пределах $150000 – $200000. Причем Вы можете увидеть границы: не менее $30000 и не более $600000.

Итак, наш менеджер получил ответ с погрешностью в $50000 в пределах наиболее достоверных значений. При планируемом обороте свыше $3000000 нас вполне устроит такая точность.

Несколько слов о самом пакете – краткое описание меню:

FILE – стандартные операции с файлами и настройка параметров печати;

EDIT – стандартные операции редактирования и встраивания внешних объектов;

FUZZY – представление числа в ячейке в виде нечеткого распределения (fuzzify) либо в виде обычного числа (crisp). Если в ячейке присутствует серый треугольник слева, значит число задано нечетко. Посмотреть его распределение можно на правом верхнем экране, редактировать - двойной щелчок на соответствующей ячейке;

FORMULA – применение функций, встроенных в пакет (как обычных, так и нечетких);

OBJECTS – операции с объектами - диаграммами нечетких распределений. Имеется полный набор операций: копирование, вырезка, вставка, печать, представление в двух- и трехмерном виде;

FORMAT – изменение графических параметров выделенных объектов: рамки, выравнивание и т.п.;

OPTIONS – общие установки таблицы;

WINDOW – стандартные операции работы с окнами;

HELP – стандартная система помощи, включающая разделы: описание меню, описание функций и работы с формулами, описание нечетких объектов, описание методов связи и встраивания внешних данных.

2.3 Пакет CubiCalc. Управление динамическими системами.

Пакет CubiCalc 2.0 фирмы HiperLogic хорошо известен специалистам, занимающимся задачами динамического управления различными ездящими, летающими и плавающими объектами. Эффективность применения CubiCalc

в данных задачах такова, что небезызвестная организация КОКОМ (США), в свое время следившая за тем, чтобы новые американские технологии в компьютерно-программной области не повышали чужой военно-промышленный потенциал, накладывала очень жесткие ограничения на распространение этого пакета [6].

Для того чтобы выяснить в чем причина того, что небольшая черная коробка с двумя дискетами и книжкой внутри совсем недавно приравнивалась к суперкомпьютерам и крылатым ракетам, рассмотрим один из примеров, поставляемых с этим пакетом.

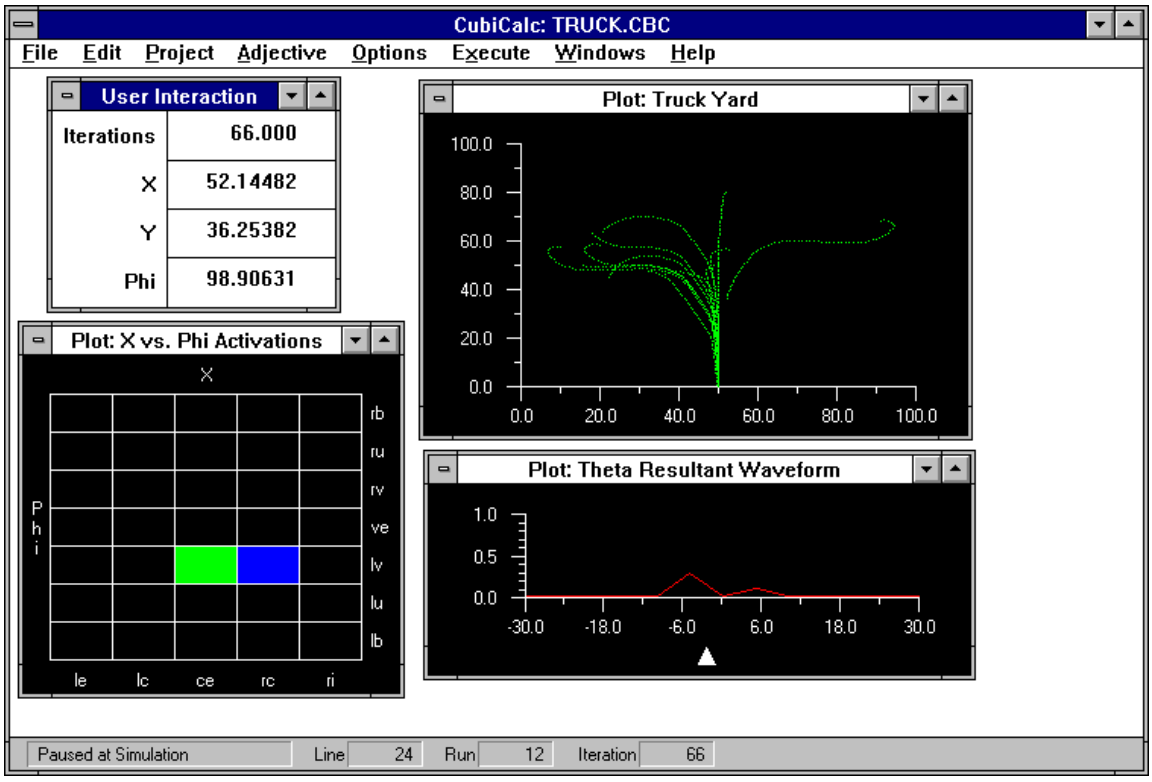

Рисунок 2.6 – Рабочая среда CubiCalc 2.0

Итак, пример, вошедший во все учебники нечеткой математики. Грузовик заезжает в гараж. Грузовик достаточно длинный, а гараж достаточно узкий.

Тот, кто хоть раз присутствовал при живописной картине маневрирования магистрального грузовика в узком дворе, может подтвердить: задача нетривиальная.

И если опустить выражения, которыми оперируют участники и очевидцы процесса, то выясняется простой факт: лучшие результаты получаются, когда на подножку грузовика встает помощник и, пользуясь критериями, вместе с водителем достигает искомого результата.

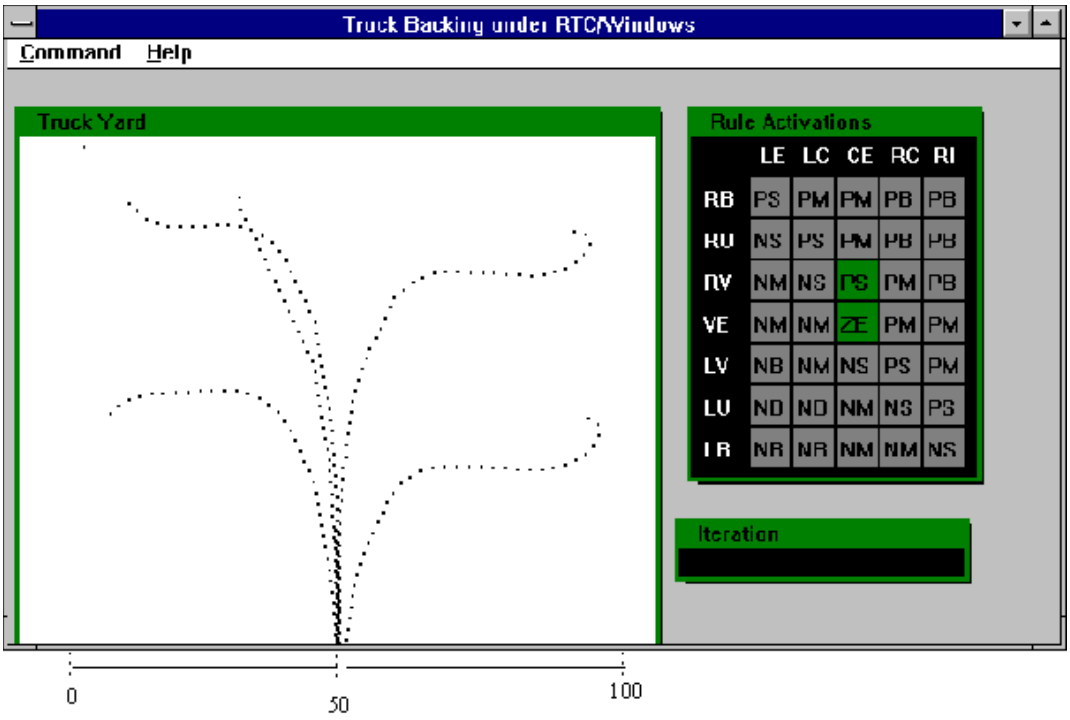

Рисунок 2.7 – Решение задачи с грузовиком

В принципе, задачу можно усложнять сколь угодно: вместо грузовика это может быть тяжелый самолет (тонн этак 70-80), вместо гаража – второй самолет (тоже не маленький), искомый результат – заправка в воздухе.

Важно то, что нас с Вами интересуют принципы, по которым может функционировать подобная система, а также то, как эти принципы свести в некоторую систему. Причем задача ставится так, что любой грузовик с любой точки двора при любых условиях должен заезжать в ворота, расположенные в любом месте двора.

Если задачу решать классическими средствами, то придется писать системы уравнений, описывающих грузовик, гараж, возможные внешние факторы, алгоритмы управления и бог знает что еще. Работа на месяц для целого отдела.

Давайте посмотрим, как такую задачу решает Cubi Calc.

Определяем переменные, которыми оперируем. Меню PROJECT/VARIABLES.

На входе замеряем «азимут грузовика относительно ворот» (в градусах, 0...360) и его «смещение относительно оси ворот» (в метрах, 0...100).

Нам важно не замерять углы с точностью до долей градуса, а сводить целевую функцию к минимуму. В данном случае целевая функция – отклонение руля, причем такое, что на въезде в ворота грузовик должен двигаться строго вдоль их оси.

Поэтому на выходе задаем переменную «поворот руля» в градусах (-30...+30).

Поскольку, как уже было сказано, нам важны принципы, для каждой переменной определим ряд нечетких распределений, соответствующим понятиям «левее-правее» для азимута, отклонения руля и смещения относительно ворот.

Меню PROJECT/ADJECTIVES FOR...

Выглядит это так (имеется в виду, что азимут 90 градусов соответствует направлению на ворота):):

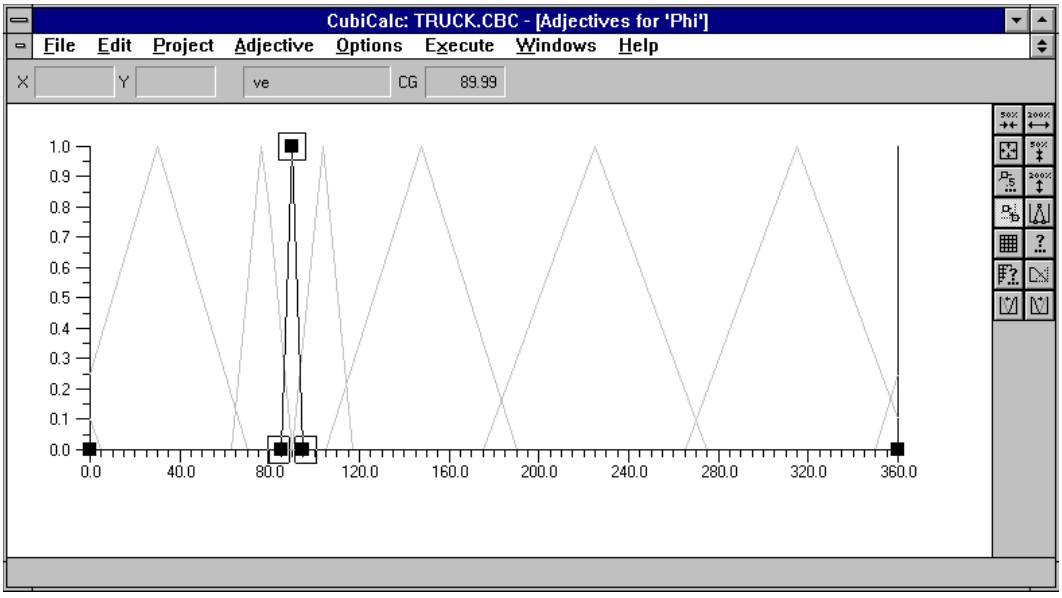

Рисунок 2.8 – Схема значений для азимута

Как видно из схемы, все значения, лежащие между 85 и 95 градусами азимута, соответствуют критерию «точно прямо» (мы полагаем, что ошибка -5...+5 градусов несущественна, т.е. сразу задаем некоторую степень точности).

Здесь же описаны распределения значений азимута, соответствующие критериям «левее, влево, глубоко влево; правее, вправо, глубоко вправо». Для каждого распределения заводится своя внутренняя (temporary) переменная – они нам понадобятся позже, для создания правил.

Для смещения относительно ворот (имеется в виду, что значения 0...49 м соответствуют «лево», 51...100м – «право». Значение 50м- соответствует критерию «точно в ворота»:

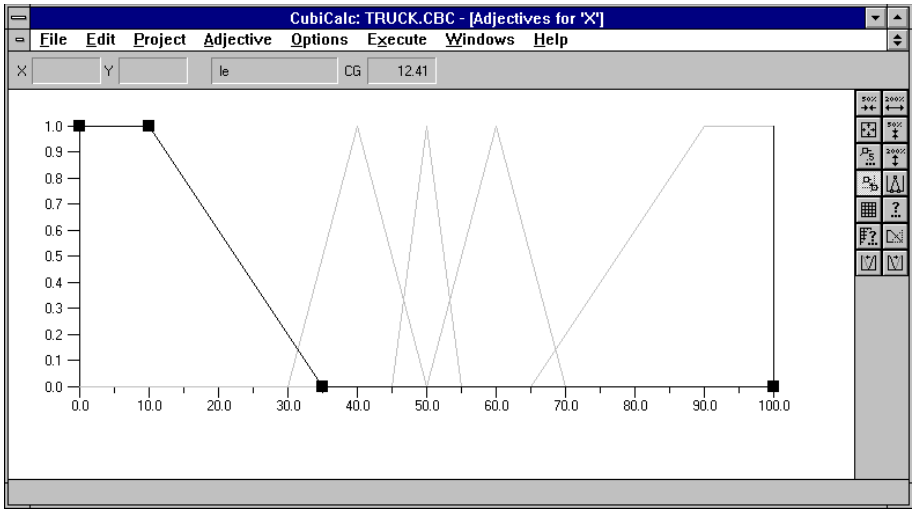

Рисунок 2.9 – Схема значений для перемещения относительно ворот

На схеме: критерий «точно в ворота» соответствует значениям между 45 и 55 м (-5...+5 м).

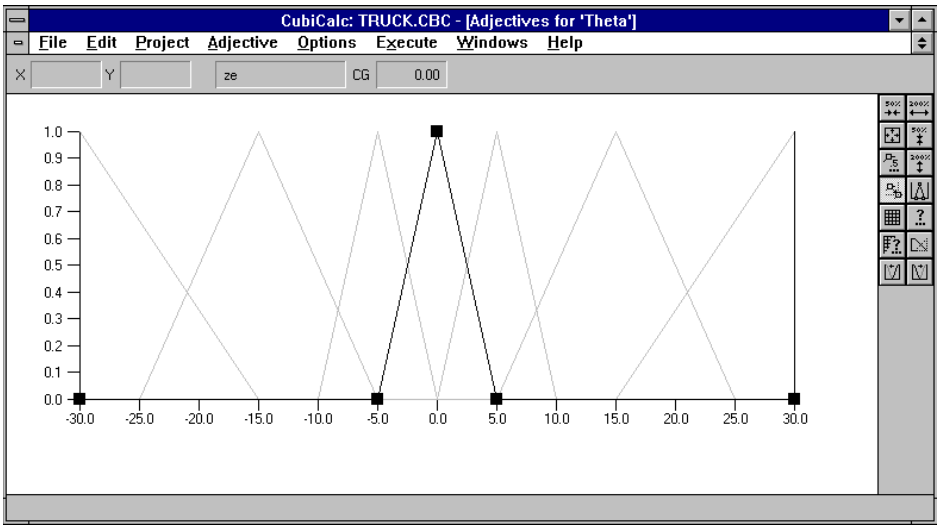

Рисунок 2.10 – Схема значений для поворота руля

Далее в меню PROJECT/VARIABLES (Рис.) мы задаем промежуточные переменные (каждому нечеткому критерию "левее", "правее" и т.п. задается своя переменная) и константы, используемые при расчетах (в данном случае - скорость);

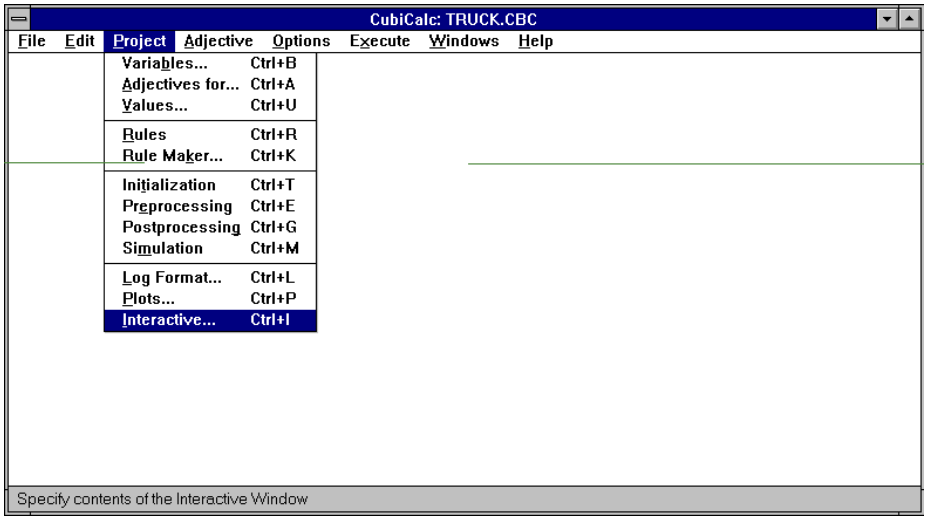

Рисунок 2.11 – Меню PROJECT

В меню PROJECT/VALUES – значения переменных.

В меню PROJECT/INITIALIZATION – инициализацию системы (параметры запуска), в данном случае – случайное расположение грузовика во дворе.

В меню PROJECT/PREPROCESSING, ../POSTPROCESSING, ../SIMULATION – соответственно, предвычисления (до работы fuzzy-части системы), поствычисления и промежуточные вычисления;

В меню PROJECT/LOG FORMAT, ../PLOTS, ../INTERACTIVE – соответственно, задание переменных, значения которых наблюдаются в процессе работы (log), построение значений указанных переменных в графическом виде, управление значениями переменных в процессе работы.

О меню PROJECT/RULES следует сказать особо. Правила (в нашем смысле – принципы) работы системы выглядят так:

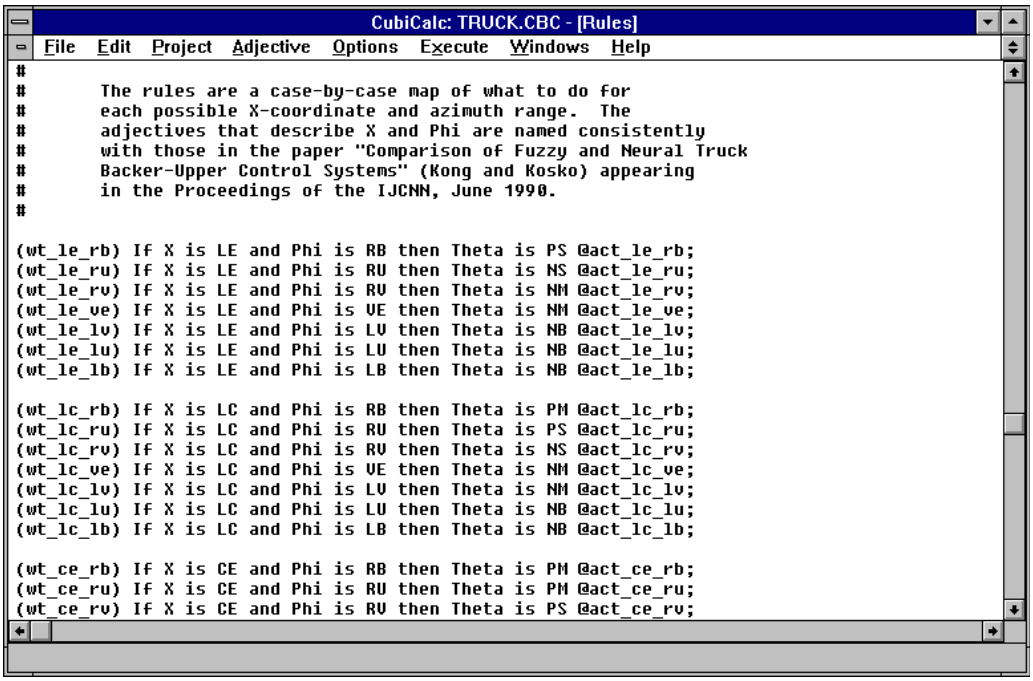

Рисунок 2.12 – Правила работы системы

Как Вы можете видеть, правила задаются в форме, интуитивно понятной пользователю:

Если переменная X имеет значение Y (и/или переменная Z имеет значение F, и/или...), то переменная A приобретает значение B, иначе переменная A приобретает... и т.д.

Если нос грузовика смотрит влево и сам грузовик находится левее относительно ворот, то надо повернуть руль глубоко вправо и продвинуться на следующий шаг.

То есть стандартная схема «если-то-иначе». С некоторым уточнением: помимо обычных логических операторов «больше», «меньше», «равно, «не равно» присутствуют нечетко логические «очень» (very), «немного» (somewhat).

Для автоматического построения правил существует опция RULEMAKER (поставляется отдельно, о ней будет сказано ниже).

Далее в меню EXECUTE Вы можете запустить полученную систему в непрерывном или циклическом режиме и вывести дополнительный экран для управления системой. В процессе работы или в пошаговом режиме Вы можете изменять различные настройки системы.

Итак, задача управления грузовиком решена. Времени на ее решение потребовалось чуть меньше, чем написать эти страницы. Разумеется, реальные задачи сложнее и разнообразнее. Причем сложность заключается в том, чтобы правильно формализовать поток данных и выделить в нем переменные.

Тем не менее с помощью Cubi Calc успешно решаются задачи динамического управления в финансовом планировании, технологические процессы, моделирование сложных предметных областей с качественно изменяющимися параметрами, сравнительно-оценочные задачи.

Следует также заметить, что нечеткая математика – как инструмент и как точка зрения, если хотите (отсюда термин fuzzy thinking), относительно молода и предоставляет специалистам широкое поле для экспериментов .

2.4 Формализация данных и выявление правил. Программа Rule Maker для Cubi Calc 2.0

Rule Maker – приложение к пакету Cubi Calc. Его назначение – обработка массивов данных, выделение в массивах групп данных по некоторым признакам (кластеризация), выявление связей между выделенными группами (построение правил). Поскольку границы кластеров задаются нечетко, возможны альтернативные выводы с указанием степени их

достоверности, то есть оперирование критериями «лучше-хуже», «возможно» и т.д [6].

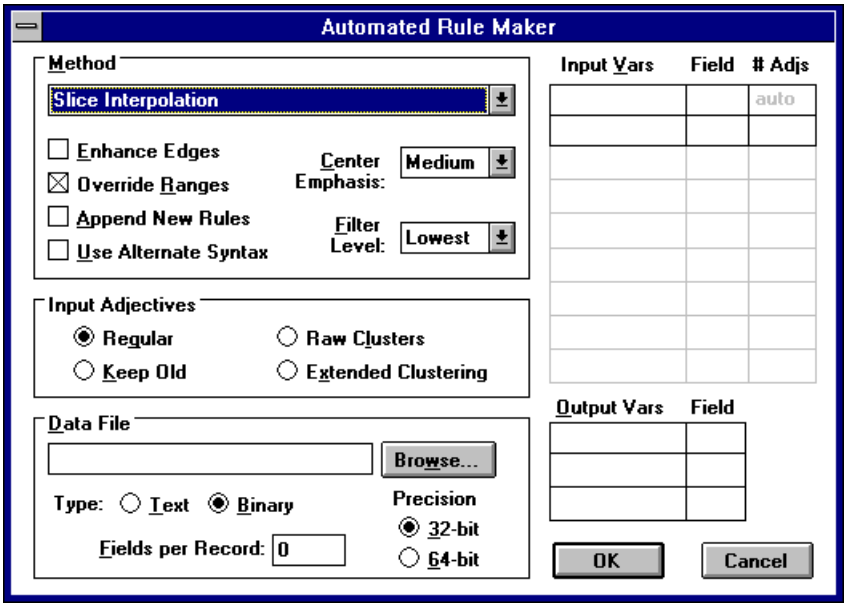

Рисунок 2.13 – Рабочая область Rule Maker

Задача нахождения правил решается в два приема:

1. Кластеризация данных (что само по себе представляет отдельную задачу). Термин "кластер" точно трактуемая как "точка концентрации". То есть под кластеризацией подразумевается нахождений областей (диапазонов значений), где соотношения "вход-выход" встречаются чаще всего. Эти диапазоны описываются как fuzzy-переменные соответственно для входов и выходов. Поскольку границы кластеров описываются нечетко, это позволяет сразу задавать некоторую точность вычислений. (см. рис на примере для двух переменных)

2. Выявление корреляционных зависимостей между входными и выходными значениями (т.н. корреляция «все по всем»). Отличие от классического метода в том, что корреляции ищутся и вычисляются не между изменениями данных, а между изменениями в границах найденных кластеров. Что и выражается в наборе правил, индицируемом в меню RULES. То есть система опять же оперирует на уровне принципов поведения объекта. В самом деле, присвоив соответствующим входным и выходным

переменным названия «больше», «меньше», «хуже», «лучше» и т.д., мы получим набор интуитивно понятных правил, описывающих поведение рассматриваемого предмета.

В меню программы можно выбрать:

Method:

correlation analysis – выявление правил с помощью вычисления корреляций;

Sparse interpolation – кусочно-линейная интерполяция функции, каждая запись в файле преобразуется в кластер, для которого определяется правило.

Slice interpolation – сглаженная интерполяция функции, вычисляются средние значения соотношений между входом и выходом;

Data Point Rules – также одна из разновидностей механизма кусочно-линейной интерполяции.

Input Vars | Field | Adjs – названия переменных, номера полей в записях, количество кластеров.

Output Vars | Field – соответственно, для выходных переменных.

Data File | Fields per record – тип файла данных (двоичный или бинарный) и количество полей на одну запись.

Input Adjectives – способы кластеризации.

В заключении стоит отметить, что Rule Maker является весьма важной компонентой Cubi Calc и может сэкономить много времени и сил для обработки больших объемов информации.

ЗАКЛЮЧЕНИЕ

Инновационные системы управления базами данных обеспечивают как физическую (самостоятельность от способа хранения и метода доступа), так и логическую независимость данных (возможность изменения одного приложения без изменения других, работающих с этими же данными).

В современные СУБД можно включать информацию, содержащую не только текст и графические изображения, а также звуковые фрагменты и даже видеоклипы.

Простота в использовании СУБД дает возможность формировать новые базы данных, используя встроенные функции, не прибегая к программированию.

При подведении итогов по данной работе приходим к такому выводу, что используя базу данных, получаем следующие преимущества:

- компактность, что позволяет избавиться от бумажных картотек и хранить всю информацию в базах данных;

- скорость. Ручная обработка занимает больше времени, чем обработка компьютером;

- низкие трудозатраты. Нет необходимости в утомительной ручной работе над данными;

- применимость. В базах данных всегда доступна свежая информация.

Другие преимущества использования базы данных – многопользовательская среда, где становится возможным осуществлять централизованное управление данными.

Для менее сложных применений вместо СУБД используются информационно-поисковые системы (ИПС), которые выполняют следующие функции:

- возможность хранить большой объем информации;

- осуществлять быстрый поиск требуемой информации;

- добавлять, удалять и изменять хранимую информацию;

- выводить ее в удобном для пользователя виде.

Показано что актуальным направлением развития электронного документооборота является переход к информационным системам, поддерживающим диалоговое взаимодействие с пользователем на естественном языке. Описана проблема преобразования словесных критериев, описывающих свойства искомых данных, в SQL-запросы, выполнение которых позволит получить требуемые информационные массивы. Математическая формализация словесных критериев фильтрации данных была формализована при помощи трапецеидальных функций принадлежности. Описан процесс преобразования трапецеидальной функции принадлежности в SQL-запросы к реляционной базе данных. Процессы преобразования математически соответствуют как теории нечетких множеств, так и концепции реляционных баз данных (т.е. стандарту SQL). Разнообразие преобразований теории нечетких множеств в дальнейшем позволит использовать широкие возможности работы с множествами (объединение, пересечение и т.д.), а соответствие требованиям SQL позволит реализовывать предложенный подход, средствами широкого класса систем управления базами данных.

Нечеткий поиск в базе данных приносит специалисту-аналитику максимальную пользу в случаях, когда требуется не только извлечь информацию, оперируя нечеткими понятиями, но и выстраивать ее по степени релевантности запроса.

СПИСОК ЛИТЕРАТУРЫ

1. Федорова Г. Н. Информационные системы; Академия - Москва, 2011. - 208 с.

2. Пирогов, В.Ю. Информационные системы и базы данных: организация и проектирование: Учебное пособие / В.Ю. Пирогов. - СПб.: БХВ-Петербург, 2009. - 528 с.

3. Круглов В.В. Интеллектуальные информационные системы: компьютерная поддержка систем нечеткой логики и нечеткого вывода/ В.В.Круглов, М.И.Дли. – М.: Физматлит, 2002. –198с

4. Конышева Л.К., Назаров Д.М. Основы теории нечетких множеств: учебное пособие. – Спб.: Питер, 2011. – 192 с.: ил.

5. Нечеткие запросы к реляционным базам данных // BaseGroups Labs [Электронный ресурс]. URL: https://basegroup.ru/community/articles/fuzzylogic-queries (дата обращения: 27.04.2016)

6. Решение задач с применением нечеткой логики// ТОРА-Центр [Электронный ресурс]. URL http://www.tora-centre.ru/library/fuzzy/fuzi_i.htm (дата обращения: 27.04.2016)

ПРИЛОЖЕНИЕ 1

ОТЗЫВ

на бакалаврскую работу студентки Ветошкиной Е.В. ″Построение информационных систем с использованием нечетких баз данных ″.

В настоящий момент информационные системы характеризуются большими размерами данных, трудной организацией и потребностью исполнять разные запросы множества пользователей. Бакалаврская работа студентки Ветошкиной Е.В. актуальна в том, что ни одна организация не обходится без баз данных, которые повышают оперативность работы с информацией. Чаще всего поступающая информация носит «нечеткий» характер и, обрабатывая ее стандартными методами и приемами, не всегда получается найти точный и конкретный ответ на интересующий нас запрос.

В бакалаврской работе выполнены все необходимые расчетно-пояснительные работы и приведены графические материалы.

В первой главе подробно описаны такие понятия, как информационные системы и базы данных: их виды, классификация и т.д. Во второй главе описаны нечеткие множества, нечеткие функции. А также приведены примеры решения задач при помощи нечеткой логики и нечетких данных.

Работа выполнена на высоком инженерном уровне, полностью соответствует заданию и требованиям, предъявляемым к выпускным работам. Оформление проекта соответствует требованиям ГОСТ и отраслевых стандартов и заслуживает отличной оценки, а дипломник Ветошкина Е.В. присвоения квалификации инженера путей сообщения по специальности автоматика, телемеханика и связь на железнодорожном транспорте.

Руководитель дипломного проекта
Доцент кафедры «Железнодорожная
автоматика, телемеханика и связь», к.т.н. Журавлев И.А.

ПРИЛОЖЕНИЕ 2

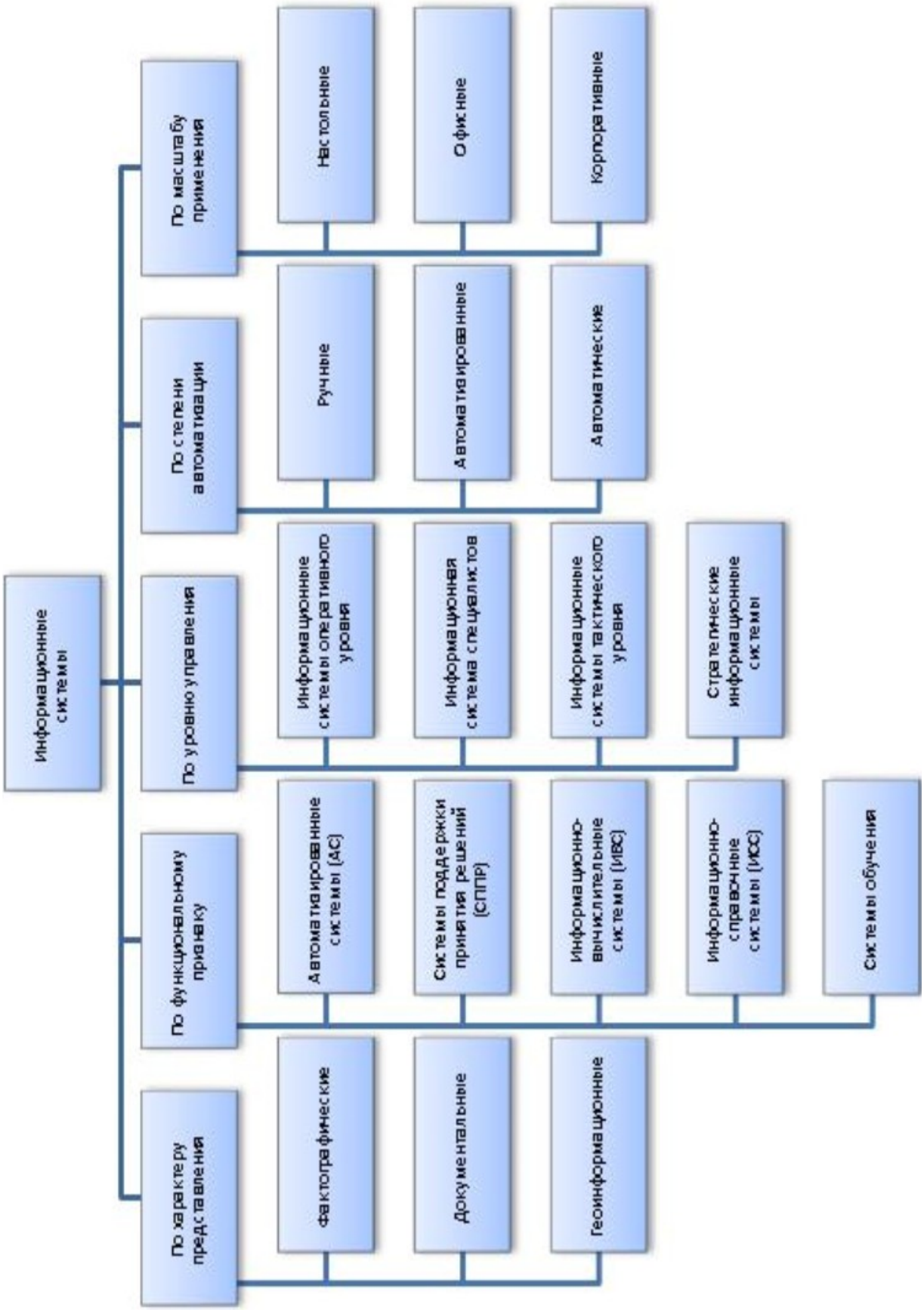

Рисунок П.1 – Классификация информационных систем