**Старооскольский технологический институт**

**(филиал)**

**Московского государственного института**

**стали и сплавов**

**(технологического университета)**

Кафедра АиПЭ

*Филатов В.А.*

Представление знаний

в информационных системах

методические указания

для выполнения лабораторных работ

для студентов специальностей

220400 – Программное обеспечение вычислительной техники и автоматических систем

**220200 – Автоматизированные системы обработки информации и управления**

**071900 – Информационные системы и технологии**

**(очная, очно-заочная, заочная формы обучения)**

***Одобрено редакционно-издательским советом института***

**Старый Оскол**

**2004**

**УДК 004**

**ББК 32.97**

**Ф 517**

**Рецензент:**

Доцент кафедры Системотехники, к.т.н. Чайников С.И.

Филатов В.А. Представление знаний в информационных системах. Методические указания к выполнению лабораторных работ. Старый Оскол, СТИ МИСиС, 2004, – 44с.

Методические указания к выполнению лабораторных работ по дисциплине «Представление знаний в информационных системах» для студентов специальностей «220400 – Программное обеспечение вычислительной техники и автоматизированных систем», «220200 – Автоматизированные системы обработки информации и управления», «071900 – Информационные системы и технологии», очной, очно-заочной, заочной форм обучения

**© Филатов В.А.**

**© СТИ МИСиС**

Содержание

[Введение 4](#_Toc90349834)

[Требования по технике безопасности 4](#_Toc90349835)

[Общее теоретическое введение 5](#_Toc90349836)

[Cодержание отчета по каждой лабораторной работе: 23](#_Toc90349837)

[Лабораторная работа №1 24](#_Toc90349838)

[1.1 Цель работы 24](#_Toc90349839)

[1.2 Теоретическая часть 24](#_Toc90349840)

[1.3 Порядок выполнения работы 34](#_Toc90349841)

[Лабораторная работа №2 35](#_Toc90349842)

[2.1 Цель работы 35](#_Toc90349843)

[2.2 Порядок выполнения работы 36](#_Toc90349844)

[Лабораторная работа №3 40](#_Toc90349845)

[3.1 Цель работы 40](#_Toc90349846)

[3.2 Порядок выполнения лабораторной работы 40](#_Toc90349847)

[Лабораторная работа №4 43](#_Toc90349848)

[4.1 Цель работы 43](#_Toc90349849)

[4.2 Порядок выполнения работы 43](#_Toc90349850)

[Список рекомендуемой литературы 43](#_Toc90349851)

# Введение

Методические указания для выполнения лабораторных работ предназначены для самостоятельного изучения основных особенностей и закрепления практических навыков при работе и инструментальным пакетом WIZWHY 3.1.

Лабораторная работа №1 посвящена приобретению первичных навыков работы в СУБД Access 97. Основное внимание уделено организации данных на этапе проектирования экспертной системы.

Лабораторная работа №2 позволяет изучить технологию работы с инструментальным пакетом WIZWHY 3.1. В лабораторной работе на конкретном примере рассмотрена методичка разработки продукционной модели экспертной системы.

Лабораторные работы №3 и 4 позволяют изучить методичку построение продукционной модели экспертной системы на основе данных находящихся в таблицах СУБД Access 97.

# Требования по технике безопасности

При работе с компьютером должны соблюдаться следующие правила:

Электрооборудование ВЦ относится к установкам напряжением до 1000 В. В помещении машинного зала основными техническими средствами, обеспечивающими безопасность работ, являются:

* заземление;
* зануление;
* отключение.

Защитным заземлением называется намеренное соединение нетоковедущих частей, которые могут случайно оказаться под напряжением, с заземляющим устройством.

Заземлению подлежат корпуса ЭВМ, трансформаторов, металлические оболочки кабелей и проводов, металлические ограждения.

Для электроустановок с напряжением до 1000 В сопротивление заземляющего устройства должно быть не менее 10 Ом.

Занулением называется намеренное соединение нетоковедущих частей, которые случайно могут оказаться под напряжением, с многократно заземленным нулевым проводом.

Защитный эффект зануления заключается в уменьшении длительности замыкания на корпус, а следовательно, в сокращении времени воздействия электрического тока на человека.

Защитное отключение — система защиты, автоматически отключающая электроустановку при возникновении опасности поражения электрическим током.

Для участка персональных компьютеров наиболее приемлемым вариантом является защитное заземление, т.к. корпуса компьютеров и периферии обычно выполнены не из токопроводящих материалов, а также имеются специальные клеммы для подключения заземления.

##### Регламентирование перерывов при работе с ПЭВМ.

При работе с разрабатываемой системой, для обеспечения оптимальной работоспособности и сохранения здоровья, на протяжении рабочей смены должны соблюдаться регламентированные перерывы.

При 8-ми часовой рабочей смене и работе на ВДТ (видеодисплейные терминалы) и ПЭВМ, регламентированные перерывы следует устанавливать через 2 часа от начала рабочей смены и через 2 часа после обеденного перерыва продолжительностью 15 минут каждый, так чтобы суммарное время перерывов было не менее 30 мин.

Продолжительность непрерывной работы с ВДТ без регламентированного перерыва не должна превышать 2 часов.

Во время регламентированных перерывов с целью снижения нервно-эмоционального напряжения, утомления зрительного анализатора, устранения влияния гиподинамии и гипокинезии, предотвращения развития познотонического утомления целесообразно выполнять комплексы упражнений.

Работающим с ПЭВМ с высоким уровнем напряженности во время регламентированных перерывов и в конце рабочего дня показана психологическая разгрузка в специально оборудованных помещениях (комната психологической разгрузки).

# Общее теоретическое введение

**1.1. Цель проведения**

Лабораторные работы выполняются закрепления теоретического материала по курсам: «Представление знаний в информационных системах» и «Интеллектуальные информационные системы»

**1.2. Организация поведения лабораторных работ**

По курсу выполняются 4 лабораторные работы.

**Лабораторная работа №1**

Изучение возможностей системы управления базами данных (СУБД) ACCESS для создания базы данных, как хранилища исходных данных для их последующего анализа и построения продукционной модели ЕСЛИ-ТО.

**Лабораторная работа №2**

Изучение возможностей системы WIZWHY – системы поиска логических правил в данных.

**Лабораторная работа №3**

Построение модели предметной области (выявление всех IF THEN правил) и анализ вероятностных характеристик этих моделей.

**Лабораторная работа №4**

Построение фрагмента экспертной системы. Анализ данных находящихся в таблицах СУБД ACCESS.

**2. Экспертные системы, их особенности**

**Применение экспертных систем**

2.1. ***Определение экспертных систем. Главное достоинство и назначение экспертных систем***

Экспертные системы (ЭС)- это яркое и быстро прогрессирующее направление в области искусственного интеллекта (ИИ). Причиной повышенного интереса, который ЭС вызывают к себе на протяжении всего своего существования является возможность их применения к решению задач из самых различных областей человеческой деятельности. Пожалуй, не найдется такой проблемной области, в которой не было бы создано ни одной ЭС или по крайней мере, такие попытки не предпринимались бы.

ЭС- это набор программ или программное обеспечение, которое выполняет функции эксперта при решении какой-либо задачи в области его компетенции. ЭС, как и эксперт-человек, в процессе своей работы оперирует со знаниями. Знания о предметной области, необходимые для работы ЭС, определенным образом формализованы и представлены в памяти ЭВМ в виде базы знаний, которая может изменяться и дополняться в процессе развития системы.

ЭС выдают советы, проводят анализ, выполняют классификацию, дают консультации и ставят диагноз. Они ориентированы на решение задач, обычно требующих проведения экспертизы человеком-специалистом. В отличие от машинных программ, использующий процедурный анализ, ЭС решают задачи в узкой предметной области (конкретной области экспертизы)на основе дедуктивных рассуждений. Такие системы часто оказываются способными найти решение задач, которые неструктурированны и плохо определены. Они справляются с отсутствием структурированности путем привлечения эвристик, т. е. правил, взятых “с потолка”, что может быть полезным в тех системах, когда недостаток необходимых знаний или времени исключает возможность проведения полного анализа.

Главное достоинство ЭС- возможность накапливать знания, сохранять их длительное время, обновлять и тем самым обеспечивать относительную независимость конкретной организации от наличия в ней квалифицированных специалистов. Накопление знаний позволяет повышать квалификацию специалистов, работающих на предприятии, используя наилучшие, проверенные решения.

Практическое применение искусственного интеллекта на машиностроительных предприятиях и в экономике основано на ЭС, позволяющих повысить качество и сохранить время принятия решений, а также способствующих росту эффективности работы и повышению квалификации специалистов.

Основными отличиями ЭС от других программных продуктов являются использование не только данных, но и знаний, а также специального механизма вывода решений и новых знаний на основе имеющихся. Знания в ЭС представляются в такой форме, которая может быть легко обработана на ЭВМ. В ЭС известен алгоритм обработки знаний, а не алгоритм решения задачи. Поэтому применение алгоритма обработки знаний может привести к получению такого результата при решении конкретной задачи, который не был предусмотрен. Более того, алгоритм обработки знаний заранее неизвестен и строится по ходу решения задачи на основании эвристических правил. Решение задачи в ЭС сопровождается понятными пользователю объяснениями, качество получаемых решений обычно не хуже, а иногда и лучше достигаемого специалистами. В системах, основанных на знаниях, правила (или эвристики), по которым решаются проблемы в конкретной предметной области, хранятся в базе знаний. Проблемы ставятся перед системой в виде совокупности фактов, описывающих некоторую ситуацию, и система с помощью базы знаний пытается вывести заключение из этих фактов (см. рис.1).

база знаний

входная

информация

механизм вывода

заключения

***Рисунок1***—Структура экспертной системы

Качество ЭС определяется размером и качеством базы знаний (правил или эвристик). Система функционирует в следующем циклическом режиме: выбор (запрос) данных или результатов анализов, наблюдения, интерпретация результатов, усвоение новой информации, выдвижении с помощью правил временных гипотез и затем выбор следующей порции данных или результатов анализов (рис.2). Такой процесс продолжается до тех пор, пока не поступит информация, достаточная для окончательного заключения.

В любой момент времени в системе существуют три типа знаний:

- Структурированные знания- статические знания о предметной области. После того как эти знания выявлены, они уже не изменяются.

- Структурированные динамические знания- изменяемые знания о предметной области. Они обновляются по мере выявления новой информации.

- Рабочие знания- знания, применяемые для решения конкретной задачи или проведения консультации.

Все перечисленные выше знания хранятся в базе знаний. Для ее построения требуется провести опрос специалистов, являющихся экспертами в конкретной предметной области, а затем систематизировать, организовать и снабдить эти знания указателями, чтобы впоследствии их можно было легко извлечь из базы знаний.

Результаты анализов и входные данные

Выбор и ввод исходных данных

наблюдения

пользователи

интерпретация

правила

гипотезы

усвоение

вывод

2.3. ***Отличительные особенности. Экспертные системы первого и второго поколения.***

1. Экспертиза может проводиться только в одной конкретной области. Так, программа, предназначенная для определения кон-

фигурации систем ЭВМ, не может ставить медицинские диагнозы.

2. База знаний и механизм вывода являются различными компонентами. Действительно, часто оказывается возможным сочетать механизм вывода с другими базами знаний для создания новых ЭС. Например, программа анализа инфекции в крови может быть применена в пульманологии путем замены базы знаний, используемой с тем же самым механизмом вывода.

3. Наиболее подходящая область применения- решение задач дедуктивным методом. Например, правила или эвристики выражаются в виде пар посылок и заключений типа “если-то”.

4. Эти системы могут объяснять ход решения задачи понятным пользователю способом. Обычно мы не принимаем ответ эксперта, если на вопрос “Почему ?” не можем получить логичный ответ. Точно так же мы должны иметь возможность спросить систему, основанную на знаниях, как было получено конкретное заключение.

5. Выходные результаты являются качественными (а не количественными).

6. Системы, основанные на знаниях, строятся по модульному принципу, что позволяет постепенно наращивать их базы знаний.

Компьютерные системы, которые могут лишь повторить логический вывод эксперта, принято относить к ЭС *первого поколения*. Однако специалисту, решающему интеллектуально сложную задачу, явно недостаточно возможностей системы, которая лишь имитирует деятельность человека. Ему нужно, чтобы ЭС выступала в роли полноценного помощника и советчика, способного проводить анализ нечисловых данных, выдвигать и отбрасывать гипотезы, оценивать достоверность фактов, самостоятельно пополнять свои знания, контролировать их непротиворечивость, делать заключения на основе прецедентов и, может быть, даже порождать решение новых, ранее не рассматривавшихся задач. Наличие таких возможностей является характерным для ЭС *второго поколения,* концепция которых начала разрабатываться 9-10 лет назад. Экспертные системы, относящиеся ко второму поколению, называют партнерскими, или усилителями интеллектуальных способностей человека. Их общими отличительными чертами является умение обучаться и развиваться, т.е. эволюционировать.

В экспертных системах первого поколения знания представлены следующим образом:

1) знаниями системы являются только знания эксперта, опыт накопления знаний не предусматривается.

2) методы представления знаний позволяли описывать лишь статические предметные области.

3) модели представления знаний ориентированы на простые области.

Представление знаний в экспертных системах второго поколения следующее:

1) используются не поверхностные знания, а более глубинные. Возможно дополнение предметной области.

2) ЭС может решать задачи динамической базы данных предметной области.

2.4. ***Области применения экспертных систем.***

Области применения систем, основанных на знаниях, могут быть сгруппированы в несколько основных классов: медицинская диагностика, контроль и управление, диагностика неисправностей в механических и электрических устройствах, обучение.

а) Медицинская диагностика.

Диагностические системы используются для установления связи между нарушениями деятельности организма и их возможными причинами. Наиболее известна диагностическая система MYCIN, которая предназначена для диагностики и наблюдения за состоянием больного при менингите и бактериальных инфекциях. Ее первая версия была разработана в Стенфордском университете в середине 70-х годов. В настоящее время эта система ставит диагноз на уровне врача-специалиста. Она имеет расширенную базу знаний, благодаря чему может применяться и в других областях медицины.

б) Прогнозирование.

Прогнозирующие системы предсказывают возможные результаты или события на основе данных о текущем состоянии объекта. Программная система “Завоевание Уолл-стрита” может проанализировать конъюнктуру рынка и с помощью статистических методов алгоритмов разработать для вас план капиталовложений на перспективу. Она не относится к числу систем, основанных на знаниях, поскольку использует процедуры и алгоритмы традиционного программирования. Хотя пока еще отсутствуют ЭС, которые способны за счет своей информации о конъюнктуре рынка помочь вам увеличить капитал, прогнозирующие системы уже сегодня могут предсказывать погоду, урожайность и поток пассажиров. Даже на персональном компьютере, установив простую систему, основанную на знаниях, вы можете получить местный прогноз погоды.

в) Планирование.

Планирующие системы предназначены для достижения конкретных целей при решении задач с большим числом переменных. Дамасская фирма Informat впервые в торговой практике предоставляет в распоряжении покупателей 13 рабочих станций, установленных в холле своего офиса, на которых проводятся бесплатные 15-минутные консультации с целью помочь покупателям выбрать компьютер, в наибольшей степени отвечающий их потребностям и бюджету. Кроме того, компания Boeing применяет ЭС для проектирования космических станций, а также для выявления причин отказов самолетных двигателей и ремонта вертолетов. Экспертная система XCON, созданная фирмой DEC, служит для определения или изменения конфигурации компьютерных систем типа VAX и в соответствии с требованиями покупателя. Фирма DEC разрабатывает более мощную систему XSEL, включающую базу знаний системы XCON, с целью оказания помощи покупателям при выборе вычислительных систем с нужной конфигурацией. В отличие от XCON система XSEL является интерактивной.

г) Интерпретация.

Интерпретирующие системы обладают способностью получать определенные заключения на основе результатов наблюдения. Система PROSPECTOR, одна из наиболее известных систем интерпретирующего типа, объединяет знания девяти экспертов. Используя сочетания девяти методов экспертизы, системе удалось обнаружить залежи руды стоимостью в миллион долларов, причем наличие этих залежей не предполагал ни один из девяти экспертов. Другая интерпретирующая система- HASP/SIAP. Она определяет местоположение и типы судов в тихом океане по данным акустических систем слежения.

д) Контроль и управление.

Системы, основанные на знаниях, могут применятся в качестве интеллектуальных систем контроля и принимать решения, анализируя данные, поступающие от нескольких источников. Такие системы уже работают на атомных электростанциях, управляют воздушным движением и осуществляют медицинский контроль. Они могут быть также полезны при регулировании финансовой деятельности предприятия и оказывать помощь при выработке решений в критических ситуациях.

е) Диагностика неисправностей в механических и электрических устройствах.

В этой сфере системы, основанные на знаниях, незаменимы как при ремонте механических и электрических машин (автомобилей, дизельных локомотивов и т.д.), так и при устранении неисправностей и ошибок в аппаратном и программном обеспечении компьютеров.

ж) Обучение.

Системы, основанные на знаниях, могут входить составной частью в компьютерные системы обучения. Система получает информацию о деятельности некоторого объекта (например, студента) и анализирует его поведение. База знаний изменяется в соответствии с поведением объекта. Примером этого обучения может служить компьютерная игра, сложность которой увеличивается по мере возрастания степени квалификации играющего. Одной из наиболее интересных обучающих ЭС является разработанная Д.Ленатом система EURISCO, которая использует простые эвристики. Эта система была опробована в игре Т.Тревевеллера, имитирующая боевые действия. Суть игры состоит в том, чтобы определить состав флотилии, способной нанести поражение в условиях неизменяемого множества правил. Система EURISCO включила в состав флотилии небольшие, способные провести быструю атаку корабли и одно очень маленькое скоростное судно и постоянно выигрывала в течение трех лет, несмотря на то, что в стремлении воспрепятствовать этому правила игры меняли каждый год.

Большинство ЭС включают знания, по содержанию которых их можно отнести одновременно к нескольким типам. Например, обучающая система может также обладать знаниями, позволяющими выполнять диагностику и планирование. Она определяет способности обучаемого по основным направлениям курса, а затем с учетом полученных данных составляет учебный план. Управляющая система может применяться для целей контроля, диагностики, прогнозирования и планирования. Система, обеспечивающая сохранность жилища, может следить за окружающей обстановкой, распознавать происходящие события (например, открылось окно), выдавать прогноз (вор-взломщик намеревается проникнуть в дом) и составлять план действий (вызвать полицию).

2.5. ***Критерий использования ЭС для решения задач.***

Существует ряд прикладных задач, которые решаются с помощью систем, основанных на знаниях, более успешно, чем любыми другими средствами. При определении целесообразности применения таких систем нужно руководствоваться следующими критериями.

1. Данные и знания надежны и не меняются со временем.

2. Пространство возможных решений относительно невелико.

3. В процессе решения задачи должны использоваться формальные рассуждения. Существуют системы, основанные на знаниях, пока еще не пригодные для решения задач методами проведения аналогий или абстрагирования (человеческий мозг справляется с этим лучше). В свою очередь традиционные компьютерные программы оказываются эффективнее систем, основанных на знаниях, в тех случаях, когда решение задачи связано с применением процедурного анализа. Системы, основанные на знаниях, более подходят для решения задач, где требуются формальные рассуждения.

4. Должен быть по крайней мере один эксперт, который способен явно сформулировать свои знания и объяснить свои методы применения этих знаний для решения задач.

В таблице один приведены сравнительные свойства прикладных задач, по наличию которых можно судить о целесообразности использования для их решения ЭС.

***Таблица 1***. Критерий применимости ЭС

|  |  |
| --- | --- |
| применимы | неприменимы |
| Не могут быть построены строгие алгоритмы или процедуры, но существуют эвристические методы решения. | Имеются эффективные алгоритмические методы. |
| Есть эксперты, которые способны решить задачу. | Отсутствуют эксперты или их число недостаточно. |
| По своему характеру задачи относятся к области диагностики, интерпретации или прогнозирования. | Задачи носят вычислительный характер. |
| Доступные данные “зашумленны”. | Известны точные факты и строгие процедуры. |
| Задачи решаются методом формальных рассуждений. | Задачи решаются прецедурными методами, с помощью аналогии или интуитивно. |
| Знания статичны (неизменны). | Знания динамичны (меняются со временем). |

В целом ЭС не рекомендуется применять для решения следующих типов задач:

- математических, решаемых обычным путем формальных преобразований и процедурного анализа;

- задач распознавания, поскольку в общем случае они решаются численными методами;

- задач, знания о методах решения которых отсутствуют (невозможно построить базу знаний).

2.6. ***Ограничения в применение экспертных систем***

Даже лучшие из существующих ЭС, которые эффективно функционируют как на больших, так и на мини-ЭВМ, имеют определенные ограничения по сравнению с человеком-экспертом.

1. Большинство ЭС не вполне пригодны для применения конечным пользователем. Если вы не имеете некоторого опыта работы с такими системами, то у вас могут возникнуть серьезные трудности. Многие системы оказываются доступными только тем экспертам, которые создавали из базы знаний.

2. Вопросно-ответный режим, обычно принятый в таких системах, замедляет получение решений. Например, без системы MYCIN врач может (а часто и должен) принять решение значительно быстрее, чем с ее помощью.

3. Навыки системы не возрастают после сеанса экспертизы.

4. Все еще остается проблемой приведение знаний, полученных от эксперта, к виду, обеспечивающему их эффективную машинную реализацию.

5. ЭС не способны обучаться, не обладают здравым смыслом. Домашние кошки способны обучаться даже без специальной дрессировки, ребенок в состоянии легко уяснить, что он станет мокрым, если опрокинет на себя стакан с водой, однако если начать выливать кофе на клавиатуру компьютера, у него не хватит “ума” отодвинуть ее.

6. ЭС неприменимы в больших предметных областях. Их использование ограничивается предметными областями, в которых эксперт может принять решение за время от нескольких минут до нескольких часов.

7. В тех областях, где отсутствуют эксперты (например, в астрологии), применение ЭС оказывается невозможным.

8. Имеет смысл привлекать ЭС только для решения когнитивных задач. Теннис, езда на велосипеде не могут являться предметной областью для ЭС, однако такие системы можно использовать при формировании футбольных команд.

9. Человек-эксперт при решении задач обычно обращается к своей интуиции или здравому смыслу, если отсутствуют формальные методы решения или аналоги таких задач.

Системы, основанные на знаниях, оказываются неэффективными при необходимости проведения скрупулезного анализа, когда число “решений” зависит от тысяч различных возможностей и многих переменных, которые изменяются во времени. В таких случаях лучше использовать базы данных с интерфейсом на естественном языке.

2.7. ***Преимущества ЭС перед человеком - экспертом***

Системы, основанные на знаниях, имеют определенные преимущества перед человеком-экспертом.

1. У них нет предубеждений.

2. Они не делают поспешных выводов.

3. Эти системы работают систематизировано, рассматривая все детали, часто выбирая наилучшую альтернативу из всех возможных.

4. База знаний может быть очень и очень большой. Будучи введены в машину один раз, знания сохраняются навсегда. Человек же имеет ограниченную базу знаний, и если данные долгое время не используются, то они забываются и навсегда теряются.

1. Системы, основанные на знаниях, устойчивы к “помехам”. Эксперт пользуется побочными знаниями и легко поддается влиянию внешних факторов, которые непосредственно не связаны с решаемой задачей. ЭС, не обремененные знаниями из других областей, по своей природе менее подвержены “шумам”. Со временем системы, основанные на знаниях, могут рассматриваться пользователями как разновидность тиражирования- новый способ записи и распространения знаний. Подобно другим видам компьютерных программ они не могут заменить человека в решении задач, а скорее напоминают орудия труда, которые дают ему возможность решат задачи быстрее и эффективнее.

6. Эти системы не заменяют специалиста, а являются инструментом в его руках.

2.8. ***История развития экспертных систем***

2.8.1. *Основные линии развития ЭС.*

Наиболее известные ЭС, разработанные в 60-70-х годах, стали в своих областях уже классическими. По происхождению, предметным областям и по преемственности применяемых идей, методов и инструментальных программных средств их можно разделить на несколько семейств.

1. **META-DENDRAL**.Система DENDRAL позволяет определить наиболее вероятную структуру химического соединения по экспериментальным данным (масс- спектрографии, данным ядерном магнитного резонанса и др.).M-D автоматизирует процесс приобретения знаний для DENDRAL. Она генерирует правила построения фрагментов химических структур.

2. **MYCIN-EMYCIN-TEIREIAS-PUFF-NEOMYCIN**. Это семейство медицинских ЭС и сервисных программных средств для их построения.

3. **PROSPECTOR-KAS. PROSPECTOR**- предназначена для поиска (предсказания) месторождений на основе геологических анализов. KAS- система приобретения знаний для PROSPECTOR.

4. **CASNET-EXPERT**. Система CASNET- медицинская ЭС для диагностики выдачи рекомендаций по лечению глазных заболеваний. На ее основе разработан язык инженерии знаний EXPERT, с помощью которой создан ряд других медицинских диагностических систем.

5. **HEARSAY-HEARSAY-2-HEARSAY-3-AGE**. Первые две системы этого ряда являются развитием интеллектуальной системы распознавания слитной человеческой речи, слова которой берутся из заданного словаря. Эти системы отличаются оригинальной структурой, основанной на использовании доски объявлений- глобальной базы данных, содержащей текущие результаты работы системы. В дальнейшем на основе этих систем были созданы инструментальные системы HEARSAY-3 и AGE (Attempt to Generalize- попытка общения) для построения ЭС.

6. Системы **AM** (Artifical Mathematician- искусственный математик) и **EURISCO** были разработаны в Станфордском университете доктором Д. Ленатом для исследовательских и учебных целей. Ленат считает, что эффективность любой ЭС определяется закладываемыми в нее знаниями. По его мнению, чтобы система была способна к обучению, в нее должно быть введено около миллиона сведений общего характера. Это примерно соответствует объему информации, каким располагает четырехлетний ребенок со средними способностями. Ленат также считает, что путь создания узкоспециализированных ЭС с уменьшенным объемом знаний ведет к тупику.

В систему AM первоначально было заложено около 100 правил вывода и более 200 эвристических алгоритмов обучения, позволяющих строить произвольные математические теории и представления. Сначала результаты работы системы были весьма многообещающими. Она могла сформулировать понятия натурального ряда и простых чисел. Кроме того, она синтезировала вариант гипотезы Гольдбаха о том, что каждое четное число, большее двух, можно представить в виде суммы двух простых чисел. До сих пор не удалось ни найти доказательства данной гипотезы, ни опровергнуть ее. Дальнейшее развитие системы замедлилось и было отмечено, что несмотря на проявленные на первых порах “математические способности”, система не может синтезировать новых эвристических правил, т.е. ее возможности определяются только теми эвристиками, что были в нее изначально заложены.

При разработке системы EURISCO была предпринята попытка преодолеть указанные недостатки системы AM. Как и в начале эксплуатации AM, первые результаты, полученные с помощью EURISCO, были эффективными. Сообщалось, что система EURISCO может успешно участвовать в очень сложных играх. С ее помощью в военно-стратегической игре, проводимой ВМФ США, была разработана стратегия, содержащая ряд оригинальных тактических ходов. Согласно одному из них, например предлагалось взрывать свои корабли, получившие повреждения. При этом корабли, оставшиеся неповрежденными, получает необходимое пространство для выполнения маневра.

Однако через некоторое время обнаружилось, что система не всегда корректно переопределяет первоначально заложенные в нее правила. Так, например, она стала нарушать строгое предписание обращаться к программистам с вопросами только в определенное время суток. Т.о., система EURISCO, так же как и ее предшественница, остановилась в своем развитии, достигнув предела, определенного в конечном счете ее разработчиком.

С 1990 года доктор Ленат во главе исследовательской группы занят кодированием и вводом нескольких сот тысяч элементов знаний, необходимых, по его мнению, для создания “интеллекту-

альной” системы. Этот проект назван Cyc (“Цик”, от английского слова enciklopaedia).

1.8.2*. Проблемы, возникающие при созданииЭС. Перспективы разработки.*

С 70-х годов ЭС стали ведущим направлением в области искусственного интеллекта. При их разработке нашли применение методы ИИ, разработанные ранее: методы представления знаний, логического вывода, эвристического поиска, распознавания предложений на естественном языке и др. Можно утверждать, что именно ЭС позволили получить очень большой коммерческий эффект от примения таких мощных методов. В этом - их особая роль.

Каталог ЭС и инструментальных программных средств для их разработки, опубликованный в США в 1987 году, содержит более 1000 систем (сейчас их уже значительно больше). В развитых зарубежных странах сотни фирм занимаются их разработкой и внедрением. Имеются и отечественные разработки ЭС, в том числе - нашедший промышленное применение.

Однако уже на начальных этапах выявились серьезные принципиальные трудности, препятствующие более широкому распространению ЭС и серьезно замедляющие и осложняющие их разработку. Они вполне естественных и вытекают из самих принципов разработки ЭС.

Первая трудность возникает в связи с постановкой задач. Большинство заказчиков, планируя разработку ЭС, в следствие недостаточной компетентности в вопросах применения методов ИИ, склонна значительно преувеличивать ожидаемые возможности системы. Заказчик желает увидеть в ней самостоятельно мыслящего эксперта в исследуемой области, способного решать широкий круг задач. Отсюда и типичные первоначальные постановки задачи по созданию ЭС: “Разработать ЭС по обработке изображения”; “Создать медицинские ЭС по лечению заболеваний опорно-двигательного аппарата у детей”. Однако, как уже отмечалось, мощность эвристических методов решения задач при увеличении общности их постановки резко уменьшается. Поэтому наиболее целесообразно (особенно при попытке создания ЭС в области, для которой у разработчиков еще нет опыта создания подобных систем) ограничиться для начала не слишком сложной обозримой задачей в рассматриваемой области, для решения которой нет простого алгоритмического способа (то есть неочевидно, как написать программу для решения этой задачи, не используя методы обработки знаний). Кроме того, важно, чтобы уже существовала сложившаяся методика решения этой задачи “вручную” или какими-либо расчетными методами. Для успешной разработки ЭС необходимы не только четкая и конкретная постановка задач, но и разработка подробного (хотя бы словесного) описания “ручного” (или расчетного) метода ее решения. Если это сделать затруднительно, дальнейшая работа по построению ЭС теряет смысл.

Вторая и основная трудность - проблема приобретения (усвоения) знаний. Эта проблема возникает при “передаче” знаний, которыми обладают эксперты-люди, ЭС. Разумеется для того, чтобы “обучить” им компьютерную систему, прежде всего требуется сформулировать, систематизировать и формализовать эти знания “на бумаге”. Это может показаться парадоксальным, но большинство экспертов (за исключением, может быть, математиков), успешно используя в повседневной деятельности свои обширные знания, испытывают большие затруднения при попытке сформулировать и представить в системном виде хотя бы основную часть этих знаний: иерархию используемых понятий, эвристики, алгоритмы, связи между ними. Оказывается, что для подобной формализации знаний необходим определенный систематический стиль мышления, более близкий математикам и программистам, чем, например, юристам и медикам. Кроме того, необходимы, с одной стороны, знания в области математической логики и методов представления знаний, с другой - знания возможности ЭВМ, из программного обеспечения, в частности, языков и систем программирования.

Таким образом, выясняется, что для разработки ЭС необходимо участие в ней особого рода специалистов, обладающих указанной совокупностью знаний и выполняющих функции “посредников” между экспертами в предметной области и компьютерными (экспертными) системами. Они получили название инженеры знаний (в оригинале - knowledge engineers), а сам процесс разработки ЭС и других интеллектуальных программ, основанных на представлении и обработке знаний - инженерией знаний (knowledge engineering). В развитых зарубежных странах специальность “инженер знаний” введена во многих вузах, в нашей стране основы инженерии знаний изучаются пока в рамках специализаций по системному программированию. Функции эксперта и инженера знаний редко совмещаются в одном лице. Чаще функции инженера знаний выполняет разработчик ЭС. Как показал опыт многих разработок, для первоначального приобретения знаний, в которых участвуют эксперты, инженеры знаний и разработчики ЭС, требуется активная работа всех трех категорий специалистов. Она может длиться от нескольких недель до нескольких месяцев.

На этапе приобретения знаний могут возникнуть трудности и психологического порядка: эксперт может препятствовать передаче своих знаний ЭС, полагая, что это снизит его престиж как специалиста и создаст предпосылки для замены его “машиной”. Однако эти опасения лишены оснований: ЭС “уверенно” работает лишь в типовых ситуациях, а также удобна в случаях, когда человек находится в состоянии стресса, в наиболее сложных ситуациях, требующих нестандартных рассуждений и оценок, эксперт- человек незаменим.

Третья серьезная трудность- в очень большой трудоемкости создания ЭС : требуется разработать средства управления базой знаний, логического вывода, диалогового взаимодействия с пользователем и т.д. Объем пограммирования столь велик, а программы столь сложны и нетрадиционны, что имеет смысл, как это принято сейчас при разработке больших программ, на первом этапе создать демонстрационный прототип системы - предварительный вариант, в котором в упрощенном виде реализованы лишь ее основные планируемые возможности и которая будет служить для заказчика подтверждниением того, что разработка ЭС для решения данной задачи принципиально возможна, а для разработчиков- основой для последующего улучшения и развития системы.

Одной из причин неудач в создании ЭС стала недооценка авторами ЭС объемов и роли неявных знаний. Системы, базы знаний которых создавались на основе справочников, в лучшем случае так справочниками и остались. Большинство же таких систем оказывались даже хуже справочников, так как сковывали исследовательскую мысль пользователя. Вторым “узким местом” ЭС оказалась модель, на которой были основаны их первые экземпляры, и лишь модель знаний, принимающая вид пороговой направленной иерархической сети с возможностью выбора в конечном из логических узлов (где каждая отдельная ситуация похожа на дерево с листьями), может стать базой для построения ЭС.

Когда стала очевидной полная непригодность этих систем и созданного для них специаллизированного аппаратного оборудования, многие обозреватели пришли к выводу, что существующая технология создания ЭС была тупиковым направлением в развитии информационных технологий. В последнее десятилетие ЭС возродились в виде систем с базой знаний, которые тесно переплетались с существующими деловыми системами. Их используют в здравоохранении, страховании, банковском деле и других областях, чтобы с помощью правил и объектовнакапливать опыт,повысить качество принимаемых решений. Базы знаний встроенысегодня в наиболее современные крупные системы. Они находятся в самой сердцевине программ- агентов, осуществляющих поиск в сети Internet, и помогают коллективам пользователей справиться с поитоками информации.

Рассмотрим факторы, стимулировавшие развитие систем с базами знаний:

- компании, добившиеся значительной экономии денежных средств благодаря технологии баз знаний, развивают и выстраивают ее в специальные бизнес- процессы, которые были бы просто невозможны без компьютерной экспертизы;

- разработаны новые технологии создания баз знаний, является необходимым средством, которое может изменить бизнес- поцесс;

- современные системы реализованы не наспециализирован-ном, а на стандартном оборудовании.

Объединение всех видов программных продуктов и их отдельных компонентов в единую ЭС признано экономически выгодным, так как прменение ЭС позволяет существенно сократить расходы на подготовку квалифицированного персонала, дальнейшую проверку работоспособности и надежности разрабатываемых и исследовательских систем, а также уменьшить время проектирования и(или) исследования.

Объектная технология, на основе которой могут создаваться и развиваться современные ЭС,- значительный шаг вперед по сравнению с CASE- средствами, т.к. она похожа на наше восприятие окружающей действительности. Наше представ- ление о моделировании меняется, то же самое происходит и с объектами, поэтому сопровождение программируемых объектов может выполнятся аналогично приспособлению наших умозрительных образов к изменению окружающих условий. Данная технология прекрасно подходит аналитикам и программистам. т.к. очень напоминает стратегию решения проблем и соответствует мыслительным процессам людей, считающихся экспертами в своей области.

Чтобы стать экспертом, специалисту нужен инструментарий, имитирующий мышление эксперта. Разработка парадигмы превращается из задачи, чуждой мышлению человека, в знакомое, привычное и легко выполняемое задание.

Как работают эксперты? Следуя принципам, заложенным в объектно- ориентированные технологии, они подразумевают проблемы на объекты или классы объектов. По мере накопления знаний в определенной области они делают обобщения, ориентируясь на выделенные объекты или классы объектов. Некоторые обобщения имеют иерархическую структуру, где свойства высших объектов наследуются объекта-

ми низшего уровня. Сущность может соответствовать нескольким классам объектов и взаимодействовать с различными объектами или классами. По мере того как знания эксперта углубляются, на их основе формируются новые ассоциации, а отдельные уровни иерархии пропадают или расширяются.

Методика объектно- ориентированного программирования основана на модели, напоминающей образы, возникающие в мозгу аналитика, которая представляет предметы и процессы в виде объектов и связей между ними. Наблюдая событие, эксперт легко выделяет знакомые образы. Для решения проблем он испытывает конкретные правила, рассматривая при этом исследуемую проблему под определенным ракурсом.

При разработке систем автоматизированного проектирования (САПР) уже нельзя обойтись без ЭС; их использование признано экономически выгодным.

С середины 80-х годов наиболее популярные системы с базами знаний создавались с ориентацией на стандартное оборудование. В этом ключ к пониманию причин успеха современной технологии баз знаний. Опыт показывает, что системы с базами знаний необходимо встраивать в самые важные бизнесс- процессы и организовывать работу персонала так, чтобы он мог максимально использовать их преимущества для достижения наилучших результатов.

**3. Структура систем, основанных на знаниях**

3.1. ***Критерий пользователя ЭС***

Структура ЭС изображена на схеме:

пользователь

эксперт +

инженер знаний

диалоговый процессор

подсистема

приобретения

знаний

база знаний

подсистема

вывода

подсистема

объяснений

Экспертные системы имеют две категории пользователей и два отдельных “входа”, соответствующих различным целям взаимодействия пользователей с ЭС:

1)обычный пользователь (эксперт), которому требуется*консультация* ЭС- диалоговый сеанс работы с ней, в процессе которой она решает некоторую экспертную задачу. Диалог с ЭС осуществляется через *диалоговый процессор*- специальную компоненту ЭС. Существуют две основные формы диалога с ЭС- диалог на ограниченном подмножестве естественного языка ( с использованием словаря- *меню* (при котором на каждом шаге диалога система предлагает выбор профессионального лексикона экспертов) и диалог на основе из нескольких возможных действий);

экспертная группа инженерии знаний, состоящая из экспертов в предметной области и инженеров знаний. В функции этой группы входит заполнение базы знаний, осуществляемое с помощью специализированной диалоговой компоненты ЭС - *подсистемы* *приобретения знаний*, которая позволяет частично автоматизировать этот процесс.

3.2.  ***Подсистема приобретения знаний***

Подсистема приобретения знаний предназначена для добавления в базу знаний новых правил и модификации имеющихся. В ее задачу входит приведение правила к виду, позволяющему подсистеме вывода применять это правило в процессе работы. В более сложных системах предусмотрены еще и средства для проверки вводимых или модифицируемых правил на непротиворечивость с имеющимися правилами.

3.3. ***База знаний***

База знаний- наиболее важная компонента экспертной системы, на которой основаны ее «интеллектуальные способности». В отличие от всех остальных компонент ЭС, база знаний- «переменная » часть системы, которая может пополняться и модифицироваться инженерами знаний и опыта использование ЭС, между консультациями (а в некоторых системах и в процессе консультации). Существует несколько способов представления знаний в ЭС, однако общим для всех них является то, что знания представлены в *символьной форме* (элементарными компонентами представления знаний являются тексты, списки и другие символьные структуры). Тем самым, в ЭС реализуется принцип *символьной природы* рассуждений, который заключается в том, что процесс рассуждения представляется как последовательность символьных преобразований.

Наиболее распространенный способ представления знаний - в виде конкретных *фактов и правил*, по которым из имеющихся фактов могут быть выведены новые. Факты представлены, например, в виде *троек*:

(АТРИБУТ ОБЪЕКТ ЗНАЧЕНИЕ).

Такой факт означает, что заданный объект имеет заданный атрибут (свойства) с заданным значением. Например, тройка (ТЕМПЕРАТУРА ПАЦИЕНТ1 37.5) представляет факт «температура больного, обозначаемого ПАЦИЕНТ1, равна 37.5». В более простых случаях факт выражается неконкретным значением атрибута, а каким либо простым *утверждением*, которое может быть *истинным* или *ложным*, например: «Небо покрыто тучами». В таких случаях факт можно обозначить каким-либо кратким *именем* (например, ТУЧИ) или использовать для представления факта сам текст соответствующей фразы.

*Правила* в базе знаний имеют вид:

ЕСЛИ А ТО S, где А- *условие*; S- *действие*. Действие S исполняется, если А истинно. Наиболее часто действие S, так же, как и условие, представляет собой *утверждение*, которое может быть *выведено* системой (то есть становится ей известной), если истинно условие правила А.

Правила в базе знаний служат для представления *эвристических знаний (эвристик)*, т.е. неформальных правил рассуждения, вырабатываемых экспертом на основе опыта его деятельности.

Простой пример правила из повседневной жизни:

ЕСЛИ небо покрыто тучами

ТО скоро пойдет дождь.

В качестве условия A может выступать либо факт(как в данном примере), либо несколько фактов A1,...,AN, соединенные логической операцией **и**:

A1 **и** A2 **и** ... **и** AN.

В математической логике такое выражение называется *коньюнкцией.* Оно считается истинным в том случае, если истинны *все* его компоненты. Пример предыдущего правила с более сложным условием:

ЕСЛИ

небо покрыто тучами **и** барометр падает

ТО

скоро пойдет дождь. (Правило 1).

Действия, входящие в состав правил, могут содержать новые факты. При применении таких правил эти факты становятся известны системе, т.е. включаются в множество фактов, которое называется рабочим множеством. Например, если факты «Небо покрыто тучами» и «Барометр падает» уже имеются в рабочем множестве, то после применения приведенного выше правила в него также включается факт «Скоро пойдет дождь».

Если система не может вывести некоторый факт, истинность или ложность которого требуется установить, то система спрашивает о нем пользователя. Например:

ВЕРНО ЛИ, ЧТО небо покрыто тучами?

При получении положительного ответа от пользователя факт «Небо покрыто тучами» включается в рабочем множество.

Существуют динамические и статические базы знаний. Динамическая база знаний изменяется со временем. Ее содержимое зависит и от состояния окружающей. Новые факты, добавляемые в базу знаний, являются результатом вывода, который состоит в применении правил к имеющимся фактам.

В системах с монотонным выводом факты, хранимые в базе знаний, статичны, то есть не изменяются в процессе решения задачи. В системах с немонотонным выводом допускается изменение или удаление фактов из базы знаний. В качестве примера системы с немонотонным выводом можно привести ЭС, предназначенную для составления перспективного плана капиталовложения компании. В такой системе по вашему желанию могут быть изменены даже те данные, которые после вывода уже вызвали срабатывание каких-либо правил. Иными словами имеется возможность модифицировать значения атрибутов в составе фактов, находящихся в рабочей памяти. Изменение фактов в свою очередь приводит к необходимости удаления из базы знаний заключений, полученных с помощью упомянутых правил. Тем самым вывод выполняется повторно для того, чтобы пересмотреть те решения, которые были получены на основе подвергшихся изменению фактов.

# Cодержание отчета по каждой лабораторной работе:

Отчет по лабораторной работе должен состоять из двух частей:

* первая часть - файл ЛАБОРАТОРНАЯ №\_\_.mdb c выполненной работой;
* вторая часть - файл в формате WORD ЛАБОРАТОРНАЯ №\_\_.doc с отчетом о выполненной работе. Отчет содержит краткие пояснения по каждому пункту задания к лабораторной работе и выводы.

# Лабораторная работа №1

## **1.1 Цель работы**

Изучение возможностей системы управления базами данных (СУБД) ACCESS для создания базы данных, как хранилища исходных данных для их последующего анализа и построения продукционной модели ЕСЛИ-ТО.

## **1.2 Теоретическая часть**

Microsoft Access как реляционная система управления базами данных

(СУБД)

Microsoft Access - это функционально полная реляционная СУБД. В ней предусмотрены все необходимые средства для определения и обработки данных, а также для управления ими при работе с большими объемами информации. Access - это не только гибкая и простая в использовании СУБД, но и система для разработки приложений баз данных, с ее помощью вы можете создать приложение, работающее в среде Windows и полностью отвечающее потребностям в управлении данными.

Определение данных и их хранение

Во время работы с документом или электронной таблицей вы обычно полностью свободны в определении содержимого документа или ячейки таблицы. На одной странице можно разместить несколько абзацев текста, таблицу, схему, несколько колонок данных. В шапке столбца электронной таблицы можно в качестве его заголовка ввести текст. Вы можете в одном столбце задавать различные числовые форматы, зависящие от назначения строк. В текстовом редакторе такая гибкость необходима для того, чтобы поместить ту или иную информацию в нужное место на странице, а в электронной таблице вы должны иметь возможность хранить исходные данные, производить необходимые вычисления и представлять результаты в нужном виде.

Эта гибкость обеспечивает успешное решение относительно небольших, хорошо сформулированных задач. Но когда электронная таблица содержит несколько сотен строк, а документы состоят из многих страниц, то работать с ними становится довольно трудно. С ростом объема данных вы можете обнаружить, что установленные электронной таблицей или текстовым редактором ограничения на память превышены или же возможности вашей компьютерной системы вообще исчерпаны. Если ваш документ предназначен для других пользователей, то вам будет трудно проконтролировать ввод новых и использование существующих данных. Если вам понадобится работать не только с цифровой или текстовой информацией, то может обнаружиться, что ваша электронная таблица не воспринимает информацию, представленную в виде рисунка или звука.

СУБД позволяет задать типы данных и способы их хранения. Вы можете определить критерии, которые СУБД будет использовать для обеспечения правильности ввода данных. В самом простом случае условие должно гарантировать, что вы случайно не введете в числовое поле буквенный символ. Другие условия могут определять область или диапазоны допустимых значений ваших данных. В наиболее совершенных системах можно задать отношения между совокупностями данных и возложить на СУБД обеспечение совместимости или целостности данных.

Microsoft Access представляет вам максимальную свободу в задании типа данных: текст, числовые значения, даты, время, денежные значения, рисунки, звук, документы. Вы можете задать форматы хранения: длина строки, точность представления чисел и даты; и представления данных для вывода на экран или печать. Чтобы быть уверенным, что в базе хранятся только правильные значения можно задать условия на значения различной степени сложности.

Так как Microsoft Access является современным приложением Windows, вы можете использовать все возможности DDE [Dynamic Data Exchange -динамический обмен данными] и OLE [Object Linking and Embedding - связь и внедрение объектов].DDE позволяет осуществлять обмен данными между Access и любым другим приложением Windows, OLE является более изощренным инструментом Windows: оно позволяет установить связь с объектами другого приложения или внедрить некоторые объекты в базу данных

Access. Это могут быть картинки, диаграммы, электронные таблицы или документы из других поддерживающих OLE приложений Windows.

Microsoft Access воспринимает множество самых разнообразных форматов данных, включая файловые структуры других СУБД. Вы можете осуществлять импорт и экспорт данных из текстовых файлов или электронных таблиц. С помощью Access вы можете непосредственно работать с файлами Paradox, dBase III, dBase IV, FoxPro и другими популярными базами данных, поддерживающими стандарт ODBC [Open Database Connectivity - открытый доступ к данным], включая Microsoft SQL Server, Oracle, DB2 и Rdb.

Архитектура Microsoft Access

Microsoft Access называет объектами все, что может иметь имя. В БД Access основными объектами являются таблицы, запросы, формы, отчеты, макросы и модули.

***Таблица*.**

Объект, который вы определяете и используете для хранения данных. Каждая таблица содержит информацию об объекте определенного типа. Таблица содержит *поля* /столбцы/ и *записи* /строки/.

Для каждой таблицы можно определить *первичный ключ* - одно или несколько полей, содержащих уникальные для каждой записи значения; и один или несколько *индексов,* помогающих ускорить доступ к данным.

***Запрос.***

Объект, который позволяет пользователю получить нужные данные из одной или нескольких таблиц. Для создания запроса можно использовать бланк QBE /запрос по образцу/ или инструкции SQL. Вы можете создать запросы на выборку, обновление, удаление или добавление данных. С помощью запросов можно создавать новые таблицы, используя данные из одной или нескольких существующих таблиц.

***Форма*.** Объект, предназначенный в основном для ввода данных, отображения их на экране или управления работой приложения. Вы можете использовать для представления пользователю данных из запросов или таблиц. Формы можно также распечатать. С помощью формы вы можете в ответ на некоторое событие, запустить макрос или процедуру VBA.

***Отчет.*** Объект, предназначенный для создания документа, который может быть впоследствии распечатан или включен в документ другого приложения.

***Макрос*.** Объект, представляющий собой структурированное описание одного или нескольких действий, которые должен выполнить Access в ответ на определенные события. Например, можно создать макрос, который в ответ на выбор некоторого элемента в основной форме открывает другую форму, или осуществляет проверку значения некоторого поля при изменении его содержимого. Из одного макроса можно запустить другой макрос или процедуру VBA.

***Модуль*.** Объект, содержащий программы, написанные на языке Visual Basic для приложений, позволяющих разбивать процесс на несколько небольших процедур. Модули могут быть самостоятельными объектами и могут быть непосредственно «привязаны» к отдельным формам или отчетам.

В таблицах хранятся данные, которые вы можете извлекать с помощью запросов. Используя формы, вы можете выводить данные на экран или изменять их. События, происходящие в формах или отчетах, могут запускать макросы. *Событие* - любое изменение состояния объекта Microsoft Access. С помощью макросов и модулей вы можете изменять ход выполнения приложения: открывать, фильтровать и изменять данные в формах и отчетах, выполнять запросы, создавать таблицы. Используя Visual Basic, вы можете создать, модифицировать и удалить любой объект Access, обрабатывать данные по строкам и столбцам.

При первом открытии окна БД Access всегда активизирует вкладку *Таблицы* и выводит на экран список доступных таблиц. В правой части окна расположены три командные кнопки:

1. **Открыть** - Нажатие этой кнопки активизирует режим таблицы для просмотра и редактирования данных.
2. **Конструктор** - Нажатие этой кнопки открывает таблицу в режиме изменения определения таблицы.
3. **Создать -** Позволяет создать новую таблицу на основе введенных данных самостоятельно либо с помощью мастера.

Когда окно базы активно, вы можете выбрать любую из этих кнопок с помощью мыши. Если вы щелкните один раз правой клавишей мыши по имени таблицы, Access выведет на экран *контекстное меню*, позволяющее выполнить ряд действий с этой таблицей.

Построение базы данных в Microsoft Access

Для создания новой таблицы окно базы данных должно быть активно, перейдите на вкладку *Таблицы* и нажмите кнопку *Создать*.

На экране появится окно диалога *Новая таблица*, выберите *Конструктор* и нажмите кнопку ОК. Access выведет окно пустой таблицы в режиме конструктора. Можно начинать определение полей. Установите курсор в первой строке столбца *Поле* и введите с клавиатуры имя первого поля. Нажмите клавишу Таb чтобы перейти в столбец Тип данных. Справа в ячейке столбца *Тип данных* появится кнопка со стрелкой вниз. Здесь и всюду в Access подобная кнопка указывает на наличие раскрывающегося списка. Нажмите эту кнопку и выберите правильное название типа данных.

Несмотря на то, что в Access при задании имен вы можете использовать символ пробела, лучше все-таки обходиться без него. Большинство баз данных SQL не поддерживает использование пробелов в именах полей. Как вы узнаете, имена полей автоматически распространяются в запросы, формы, отчеты, создаваемые на основе этих таблиц.

После того, как вы выберете тип данных, Access выведет свойства поля в нижней части окна. Здесь вы можете задать значение свойств поля, причем некоторые из значений будут устанавливаться по умолчанию. В столбце *Описание* вы можете ввести пояснительный текст для каждого из полей. Во всех случаях при выборе поля в запросе или в форме Access выводит его описание в строке состояния в нижней части окна.

**Типы данных**.

Access поддерживает восемь основных типов данных, кроме того предоставляет мастера подстановок, который поможет вам задать характеристики чужих ключей, используемых при установлении связей с другими таблицами.

|  |  |  |
| --- | --- | --- |
| Тип данных | Использование | Размер |
| Текстовый | Алфавитно-цифровые данные. | До 255 байтов |
| Поле МЕМО | Алфавитно-цифровые данные – предложения, абзацы, тексты | До 64000 байтов |
| Числовой | Числовые данные | 1,2,.4 или 8 байтов  16 байтов для кода реплики |
| Дата/Время | Даты и время | 8 байтов |
| Денежный | Данные о денежных суммах, хранящиеся с 4 знаками после запятой | 8 байтов |
| Счетчик | Уникальное длинное целое, генерируемое при создании новой записи | 4 байта |
| Логический | Логические данные | 1 бит |
| Поле объекта OLE | Картинки, диаграммы и др. объекты из приложений Windows | До 1 гигабайта |

Для символьных данных выбирают *Текстовый* тип и указывают максимальную длину поля. *Поле МЕМО* следует использовать только в тех случаях, когда размер текста может превысить 255 символов, или когда в тексте встречаются символы форматирования, такие как табуляция или возврат каретки. Если вы выбрали *Числовой* тип, вам следует тщательно обдумать вопрос о значении свойства Размер поля. Тип *Дата/Время* используется для хранения календарных дат и позволяет выполнять вычисления в единицах измерения времени /минутах, секундах, часах, днях, годах/. *Денежный* тип имеет ту же точность, что и тип Числовой, но с фиксированным числом знаков после запятой. *Счетчик* специально предназначен для автоматической генерации значений первичного ключа. Таблица не может содержать более одного поля с типом данных Счетчик. *Логический* тип данных используется для хранения значений Истина и Ложь. Он особенно полезен, если вам нужно отметить оплачены ли счета, выполнены ли тесты и т. п. Тип *Поле объекта OLE* позволяет хранить и редактировать документы Microsoft Word, электронные таблицы Microsoft Excel, картинки Microsoft Power Point, звуковые файлы, видеофайлы.

**Свойства полей**

*Свойства на вкладке Общие:*

* Размер поля. Поле с текстовым типом данных может иметь размер от 1 до 255 символов. По умолчанию устанавливается 50 символов. Для числового типа данных размер поля может быть следующим:

|  |  |
| --- | --- |
| Байт | Целые числа от 0 до 255, занимает при хранении 1 байт |
| Целое | Целые числа от –32768 до +32767, занимает 2 байта |
| Длинное целое | Целые числа от –2147483648 до +2147483647, занимает 4 байта |
| С плавающей точкой/4 байта/ | 38 38  Числа с точностью до 6 знаков от –3,4х10 до+3,4х10, занимает 4 байта |
| С плавающей точкой/8байт/ | 308 308  Числа с точностью до 10 знаков от –1,797х10 до +1,97х10, занимает 8 байт |
| Код репликации | 16-байтовый Globally Unique Identifier (GUID) |

* Формат поля. Вы можете задать формат представления данных при выводе на экран или печать. Для типов данных *Текстовый и Поле МЕМО*  вы можете задать специальные форматы. Для типов данных *Числовой*, *Денежный, Счетчик* существует стандартный набор форматов поля, приведенный ниже:

|  |  |
| --- | --- |
| Основной формат | Устанавливается по умолчанию / отсутствуют разделители групп разрядов, число десятичных знаков зависит от точности данных/ |
| Денежный | Символы денежной единицы и два знака после запятой |
| Фиксированный | По крайней мере, один знак до и два после запятой |
| С разделителями разрядов | Два знака после запятой и разделители групп разрядов |
| Процентный | Процент |
| Экспоненциальный | 3  Экспоненциальная запись /например 1,05х10/ |

Для типа данных *Дата/Время* существует следующий набор форматов:

|  |  |  |
| --- | --- | --- |
|  | Дата | Время |
| Полный формат | Устанавливается по умолчанию 15.04.99 05:30:10 РМ |  |
| Длинный формат | Среда 15 Апрель 1999 | 17:30:10 |
| Средний формат | 15-апр-99 | 05:30 РМ |
| Краткий формат | 15.04.99 | 17:30 |

Для логического типа данных используются следующие форматы:

Да/Нет Устанавливается по умолчанию

Истина/Ложь

Вкл/Выкл

* Число десятичных знаков. Для числового и денежного типов вы можете задать число знаков после запятой. По умолчанию устанавливается значение *Авто*, при котором для *денежного, фиксированного, с разделителями* *разрядов и процентного* форматов поля выводятся два десятичных знака после запятой, а для *основного* формата число десятичных знаков определяется текущей точностью числовых значений. Вообще оно может изменяться от 1 до 15.
* Маска ввода. Для типов данных *Текстовый, Числовой, Денежный* и *Дата/Время* можно задать маску, которую пользователь увидит при вводе данных в поле.
* Подпись. Вы можете определить более содержательное название поля, которое Access будет выводить во всех элементах управления форм и в заголовках отчетов.
* Значение по умолчанию. Вы можете определить значение по умолчанию для всех типов данных, кроме *Счетчика и Поля объекта OLE* . Для числового типа значением по умолчанию является 0, а для текстового и МЕМО – Null.
* Условие на значение. Можно задать выражение, которое при вводе или редактировании значения поля всегда должно быть истинным. Например, <100 означает, что значение поля должно быть меньше 100. Кроме того, можно проверять совпадение значения поля с одним из нескольких допустимых значений, задавать выражения, содержащие любые встроенные в Access функции.
* Сообщение об ошибке. Вы можете определять текст сообщения, которое Access будет выводить в случаях, когда вводимое значение не удовлетворяет условию на значение.
* Обязательное поле. Если вы не допускаете, чтобы в этом поле хранилось значение Null , установите для этого свойства значение *Да.*
* Пустые строки. Для текстовых и МЕМО полей можно разрешить ввод пустых строк. /Более подробно ниже/.
* Индексированное поле. Чтобы ускорить доступ к данным, задайте построение индекса для полей с типом данных Текстовый, Числовой, Денежный, Дата/Время и Счетчик. Можно также указать, чтобы значения, хранящиеся в индексированном поле, не повторялись.

*Свойства на вкладке Подстановка*:

* Тип элемента управления. Задает тип элемента управления, используемого по умолчанию для отображения данного поля в формах, отчетах и объектах в режиме таблицы. Для большинства полей выбирайте тип *Поле.* Если поле является чужим ключом, выберите *Список* или *Поле* со списком для отображения значений из связанной таблицы.
* Тип источника строк. Если для свойства Тип элемента управления установлено значение *Список* или *Значение со списком*, это свойство позволяет задать тип источника данных для элемента управления: *Таблица/запрос, Список значений или Список полей.*
* Источник строк. Если для свойства Тип источника строк установлено значение Таблица/запрос, укажите таблицу или запрос, из которого поступают значения для списка.
* Присоединенный столбец. Значение указанного столбца списка используется в качестве значения элемента управления. Если источник данных содержит один столбец, установите значение 1.
* Число столбцов. Введите число столбцов, выводящихся в списке или в раскрывающемся списке поля со списком.
* Заглавия столбцов. При установке значения *Да* значения свойства Подпись, заданные в источнике данных, выводятся в качестве заголовков столбцов списка.
* Ширина столбцов. Введите значения для ширины столбцов, разделяя их точкой с запятой. Если вы не хотите выводить некоторый столбец, укажите для него ширину, равную 0.
* Число строк списка. Если элементом управления является поле со списком, это свойство определяет число строк, выводимых в списке.
* Ширина списка. Определяет ширину раскрывающегося списка для поля со списком. Значением по умолчанию является *Авто*, при котором ширина раскрывающегося списка равна ширине самого элемента управления. Если ширина поля недостаточна, введите подходящее значение.
* Ограничиться списком. При установке значения *Да* в поле можно ввести только значения, представленные в списке. Чтобы иметь возможность вводить произвольные значения, установите для данного свойства значение *Нет*.

**Задание простых условий на значение для полей.**

Чтобы задать простое условие на значение, введите выражение в строке свойства *Условие на значение* для этого поля. Access не позволит ввести данные, не удовлетворяющие этому условию. Условие на значение задается выражением, которое в общем случае состоит из операторов сравнения и значений /операндов/. Вы можете использовать несколько сравнений, связанных логическими операторами OR /ИЛИ/ и AND /И/. Текстовые значения должны всегда заключаться в кавычки. Если в качестве операнда используется дата, она должна быть заключена в символы числа /#/, например, #15.04.99#. В качестве операторов используются символы сравнения. Например, вы хотите, чтобы значение числового поля всегда было меньше 1000. Для этого вам следует ввести выражение <1000. Например, чтобы числовое значение находилось в интервале от 50 до 100, вам следует ввести >50 And<100 или BETWEEN 50 And 100.

Некоторые операторы сравнения, используемые при задании условия на значение:

|  |  |
| --- | --- |
| Оператор | Описание |
| < | Меньше |
| <= | Меньше или равно |
| > | Больше |
| >= | Больше или равно |
| = | Равно |
| <> | Не равно |
| IN | Проверяет на равенство любому значению из списка, операндом является список, заключенный в круглые скобки |
| BETWEEN | Проверяет, что значение поля находится в заданном диапазоне, верхняя и нижняя границы диапазона разделяются логическим оператором And |
| LIKE | Проверяет соответствие текстового или МЕМО поля заданному шаблону символов |

Если вам потребуется проверить соответствие значения текстового или МЕМО поля заданному шаблону, вы можете воспользоваться оператором сравнения LIKE. В качестве операнда задайте символьную строку, используя символы шаблона, приведенные ниже:

|  |  |
| --- | --- |
| Символ шаблона | Описание |
| ? | Заменяет один произвольный символ |
| \* | Заменяет любое, включая нулевое, количество произвольных символов, используется для замены последовательностей символов |
| # | Заменяет одну цифру |

**Задание маски ввода**

Чтобы облегчить ввод форматированных данных, Access позволяет задать маску ввода для поля с любым типом данных, кроме *Поля МЕМО, Счетчика* и *Поля объекта OLE*. Маска ввода определяется с помощью символов, приведенных ниже. Маска ввода состоит из трех частей, разделенных точкой с запятой. Первая часть – собственно маска, состоящая из символов маски и постоянных символов. Необязательная вторая часть указывает Access, нужно ли сохранять постоянные символы маски в этом поле. Если они включаются в значение поля, укажите во второй части 0, если сохраняться будут только введенные пользователем символы, укажите 1.

Символы, используемые для задания маски ввода:

|  |  |
| --- | --- |
| Символ маски | Описание |
| 0 | В данную позицию должна быть введена цифра, не допускается ввод знаков плюс + и минус -. |
| 9 | В данную позицию может быть введена цифра или пробел. Если пользователь пропустит эту позицию, Access не занесёт в эту позицию никакой информации. |
| # | В данную позицию может быть введена цифра, пробел, плюс, минус. Если пользователь пропустит эту позицию, Access занесёт в нее пробел. |
| L | В данную позицию должна быть введена произвольная буква. |
| ? | В данную позицию может быть введена буква. Если пользователь пропустит ее, Access не занесёт никакой информации. |
| A | В данную позицию должна быть введена буква или цифра. |
| А | В данную позицию может быть введена буква или цифра. Если пользователь пропустит её, Access не занесёт никакой информации. |
| & | В данную позицию должен быть введен произвольный символ или пробел. |
| C | В данную позицию может быть введен любой символ или пробел. Если пользователь пропусти её, Access не занесет никакой информации. |
| . | Место для десятичной запятой /зависит от установок в окне Язык и стандарты панели управления Windows/. |
| , | Разделитель групп разрядов /зависит от установок в окне Язык и стандарты панели управления Windows/. |
| :-/ | Разделители компонентов даты и времени /зависят от установок в окне Язык и стандарты панели управления Windows/. |
| < | Преобразует все символы справа к нижнему регистру. |
| > | Преобразует все символы справа к верхнему регистру. |
| ! | Указывает, что маску надо заполнять справа налево. |
| \ | Указывает, что следующий символ следует рассматривать в качестве постоянного символа, даже если он является символом маски. |
| «литерал» | Вместо того, чтобы многократно использовать символ «\», вы можете заключить любой литерал в двойные кавычки. |

**Создание первичного ключа**

Каждая таблица в реляционной базе должна иметь первичный ключ. Создать его в Access очень просто. Для выбора первого ключевого поля таблицы в окне конструктора щелкните в области выделения, расположенной слева от имени поля. Если вы хотите включить в первичный ключ несколько полей, нажмите клавишу Ctrl и, не отпуская ее, щелкните по области выделения полей. Затем нажмите кнопку *Ключевое поле* на панели инструментов или выберите команду *Правка->Ключевое поле*. В подтверждение того, что ключ задан, Access выведет в области выделения слева от каждого поля, символ ключа. Остался всего один шаг – сохранить созданную таблицу. Выберите команду *Файл ->Сохранить*. Access откроет окно диалога *Сохранение,* введите имя таблицы и нажмите кнопку ОК.

**Задание условия на значения для таблицы**

Условие на значение для поля проверяется при вводе в это поле нового значения, а условие на значение для таблицы проверяется при сохранении или добавлении записи. В режиме конструктора нажмите кнопку *Свойства* на панели инструментов или выберите команду *Вид ->Свойства*, в результате чего откроется окно *Свойства таблицы*. Первая строка предназначена для ввода описания таблицы. Во вторую строку вы можете ввести любое допустимое выражение. Обратите внимание, что в условии на значение для таблицы можно сравнить содержимое одного поля с содержимым другого. Третья строка предназначена для сообщения, выводимого при нарушении этого условия на значения. Свойство *Фильтр* позволяет задать условие отбора записей, которые будут выводиться на экран в режиме таблицы. В строке *Порядок* сортировки можно указать поля, по которым будет производиться сортировка записей в режиме таблицы. По умолчанию Access выводит записи в порядке возрастания значений первичного ключа.

**Определение связей**

После того, как вы определите несколько таблиц, вам нужно сообщить Access, как они связаны друг с другом. Создать связь типа «многие-ко-многим» можно только с помощью дополнительной связующей таблицы. В неё необходимо включить поля, характеризующие конкретные отношения. Такую таблицу несложно создать в режиме конструктора. Чтобы определить связи, вернитесь в окно базы данных, выберите команду *Сервис->Схема данных*. Если вы впервые определяете связи в этой базе данных, Access откроет пустое окно *Схема* *данных*, а затем выведет на экран окно диалога *Добавление таблицы*. В этом окне выделите все таблицы и нажмите кнопку *Добавить*. Затем нажмите кнопку *Закрыть*.

## **1.3 Порядок выполнения работы**

1. Создать таблицу ЗАДАНИЕ\_1 по следующей структуре:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Табельный номер | Дата рождения | Фамилия | Оклад | Отдел | Отработано дней | Премия | Пол |
| **число** | **дата** | **текст** | **число** | **Текст** | **Число** | **число** | **текст** |

1. Ввести произвольные данные в таблицу (30 строк), учитывая следующие ограничения целостности:

* варианты окладов 400, 500, 650, 800 рублей;
* варианты отделов ВЦ, АСУ, БУХГАЛТЕРИЯ
* варианты премий 500,600,700 рублей;
* количество мужчин и женщин определите самостоятельно.

1. Выберите два атрибута таблицы и задайте условий на вводимые значения.
2. Обязательно используйте такое свойство полей как «Маска ввода».
3. Обоснуйте и выберите ключевой атрибут в создаваемой таблице.

Таким образом, Вы сформируете свою индивидуальную таблицу, состоящую из 30 строк.

Сделайте распечатку созданной Вами таблицы.

1. Создать три справочника (новые таблицы):

* СПРАВОЧНИК\_ОКЛАДЫ;
* СПРАВОЧНИК\_ПРЕМИИ;
* СПРАВОЧНИК\_ОТДЕЛЫ.

1. Для атрибутов «ОКЛАД», «ПРЕМИЯ», «ОТДЕЛЫ» создать **поле со списком** из соответствующих справочников.
2. Пользуясь интерфейсом СУБД «ACCESS» - “СЕРВИС-СХЕМА ДАННЫХ” создать схему данных.

# Лабораторная работа №2

## **2.1 Цель работы**

Изучение возможностей системы WIZWHY – системы поиска логических правил в данных.

**Назначение системы**

Система WIZWHY на сегодняшний день является одним из лидеров на рынке продуктов DATA MINING. Она позволяет:

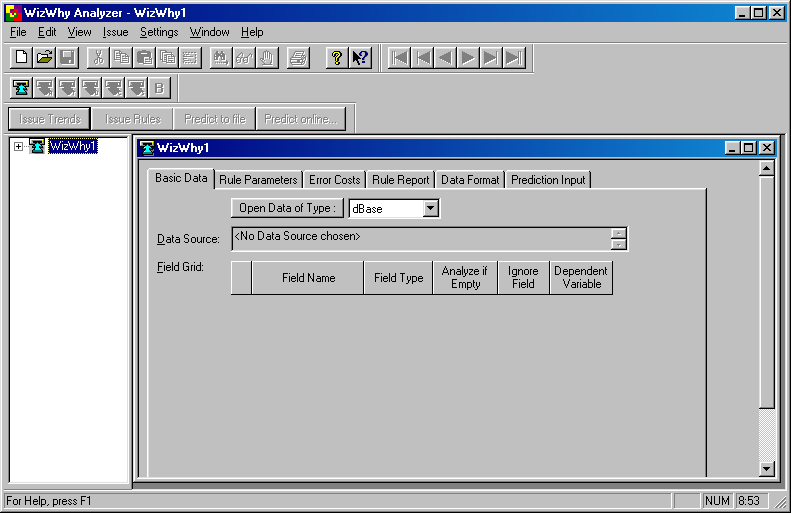
1. Формировать IF THEN правила по анализируем данным;
2. Вычислять вероятность ошибки для каждого правила;
3. Вычислять ошибки прогнозов;
4. Использовать сформированные правила для задачи прогнозирования.

## **2.2 Порядок выполнения работы**

**Загрузка и управления данными.**

1. Загрузить систему.

В качестве примера используем файл данных по ультразвуковому зондированию почек:



## 

#### *Таблица №1*

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Diagnosis | Age | Sex | LR | Length | Width | Thickness | Thickpar | Speed | Index | Accel |
| 1 | 62 | F | R | 127 | 52 | 43 | 14 | 13.3 | 0.698 | 140 |
| 1 | 43 | M | L | 103 | 44 | 49 | 15 | 16.3 | 0.634 | 291 |
| 1 | 58 | M | R | 103 | 58 | 46 | 17 | 16.5 | 0.704 | 143 |
| 1 | 37 | M | L | 112 | 53 | 51 | 18 | 18.2 | 0.562 | 189 |
| 1 | 21 | M | L | 126 | 62 | 45 | 14 | 18.5 | 0.613 | 116 |
| 1 | 74 | M | R | 115 | 57 | 49 | 16 | 19.1 | 0.69 | 85 |
| 1 | 62 | M | R | 103 | 66 | 45 | 16 | 19.2 | 0.657 | 65 |
| 1 | 43 | M | R | 104 | 54 | 46 | 14 | 19.3 | 0.574 | 629 |
| 1 | 34 | F | L | 110 | 52 | 42 | 19 | 19.3 | 0.686 | 152 |
| 1 | 68 | F | R | 112 | 52 | 42 | 17 | 19.4 | 0.593 | 258 |
| 1 | 37 | M | R | 119 | 51 | 41 | 14 | 19.8 | 0.65 | 101 |
| 1 | 38 | M | L | 107 | 59 | 56 | 12 | 20.9 | 0.572 | 279 |
| 1 | 67 | F | R | 113 | 57 | 47 | 17 | 20.9 | 0.681 | 115 |
| 1 | 46 | F | L | 107 | 60 | 36 | 15 | 21.8 | 0.678 | 352 |
| 1 | 67 | F | R | 135 | 70 | 67 | 27 | 24 | 0.583 | 51 |
| 1 | 52 | M | L | 111 | 59 | 50 | 21 | 26.6 | 0.644 | 379 |
| 1 | 42 | M | R | 82 | 47 | 42 | 14 | 26.8 | 0.651 | 538 |
| 1 | 47 | F | L | 114 | 63 | 51 | 17 | 27.8 | 0.645 | 589 |
| 1 | 23 | F | L | 128 | 39 | 56 | 21 | 30.6 | 0.62 | 255 |
| 1 | 35 | F | L | 114 | 50 | 41 | 14 | 31.2 | 0.622 | 445 |
| 1 | 64 | F | R | 97 | 57 | 39 | 16 | 34.7 | 0.68 | 368 |
| 1 | 56 | M | L | 125 | 74 | 72 | 2 | 46.6 | 0.638 | 685 |
| 2 | 46 | F | R | 112 | 68 | 88 | 18 | 2.3 | 0.584 | 459 |
| 2 | 58 | M | L | 129 | 67 | 58 | 18 | 12.5 | 0.686 | 151 |
| 2 | 69 | F | R | 115 | 69 | 44 | 14 | 13.2 | 0.657 | 231 |
| 2 | 69 | M | L | 126 | 59 | 49 | 13 | 14.1 | 0.652 | 282 |
| 2 | 54 | M | R | 98 | 87 | 41 | 24 | 14.3 | 0.637 | 352 |
| 2 | 67 | M | R | 111 | 59 | 47 | 18 | 14.6 | 0.742 | 242 |
| 2 | 70 | F | R | 108 | 58 | 37 | 11 | 14.6 | 0.663 | 139 |
| 2 | 67 | M | L | 129 | 64 | 58 | 17 | 15.2 | 0.693 | 382 |
| 2 | 55 | M | L | 125 | 59 | 48 | 18 | 15.4 | 0.674 | 330 |
| 2 | 65 | F | R | 111 | 54 | 51 | 13 | 15.4 | 0.727 | 257 |
| 2 | 70 | F | L | 108 | 65 | 51 | 14 | 16.3 | 0.724 | 250 |
| 2 | 63 | F | R | 121 | 67 | 55 | 16 | 16.4 | 0.653 | 148 |
| 2 | 56 | F | R | 99 | 55 | 47 | 16 | 16.8 | 0.693 | 103 |
| 2 | 60 | F | R | 105 | 56 | 46 | 14 | 17.7 | 0.526 | 219 |
| 2 | 54 | M | L | 107 | 57 | 43 | 14 | 17.9 | 0.651 | 254 |
| 2 | 48 | F | R | 100 | 56 | 44 | 16 | 18 | 0.667 | 114 |
| 2 | 74 | M | L | 104 | 56 | 56 | 16 | 18.2 | 0.643 | 88 |
| 2 | 62 | F | L | 118 | 56 | 41 | 14 | 18.3 | 0.611 | 347 |
| 2 | 60 | F | L | 108 | 56 | 50 | 14 | 18.6 | 0.546 | 216 |
| 2 | 63 | F | L | 110 | 61 | 56 | 14 | 19 | 0.645 | 216 |
| 2 | 64 | M | R | 123 | 61 | 57 | 20 | 19.1 | 0.632 | 173 |
| 2 | 54 | M | R | 105 | 58 | 43 | 16 | 21 | 0.62 | 230 |
| 2 | 47 | F | L | 116 | 73 | 56 | 17 | 21.4 | 0.636 | 178 |
| 2 | 47 | F | L | 109 | 58 | 48 | 11 | 21.5 | 0.544 | 266 |
| 2 | 66 | F | R | 98 | 64 | 56 | 19 | 22.3 | 0.655 | 111 |
| 2 | 67 | F | L | 103 | 44 | 42 | 14 | 22.6 | 0.682 | 656 |
| 2 | 54 | M | L | 119 | 50 | 48 | 20 | 23.5 | 0.65 | 188 |
| 2 | 70 | F | R | 105 | 56 | 41 | 13 | 23.5 | 0.663 | 242 |
| 2 | 66 | F | L | 111 | 61 | 50 | 16 | 24.7 | 0.689 | 189 |
| 2 | 56 | F | L | 99 | 58 | 47 | 16 | 25 | 0.686 | 196 |
| 2 | 47 | F | R | 99 | 62 | 50 | 12 | 26.2 | 0.544 | 235 |
| 2 | 69 | F | L | 125 | 66 | 51 | 18 | 26.2 | 0.667 | 275 |
| 2 | 37 | M | L | 120 | 89 | 61 | 16 | 27.9 | 0.566 | 218 |
| 2 | 48 | F | L | 113 | 77 | 49 | 22 | 30.5 | 0.686 | 210 |
| 2 | 45 | F | L | 115 | 56 | 52 | 17 | 34.2 | 0.587 | 410 |
| 2 | 70 | F | L | 108 | 48 | 42 | 19 | 36.1 | 0.69 | 219 |
| 2 | 67 | F | L | 122 | 72 | 64 | 20 | 43.3 | 0.674 | 229 |
| 3 | 42 | M | L | 104 | 69 | 51 | 24 | 14.5 | 0.729 | 211 |
| 3 | 21 | M | R | 144 | 101 | 49 | 6 | 16.3 | 0.707 | 194 |
| 3 | 67 | F | L | 99 | 52 | 52 | 8 | 16.3 | 0.744 | 332 |
| 3 | 34 | F | R | 105 | 54 | 46 | 19 | 19.4 | 0.704 | 98 |
| 3 | 47 | F | R | 116 | 85 | 64 | 24 | 19.5 | 0.804 | 416 |
| 3 | 38 | M | R | 118 | 45 | 60 | 12 | 19.9 | 0.701 | 354 |
| 3 | 56 | M | R | 118 | 55 | 43 | 22 | 21 | 0.735 | 165 |
| 3 | 37 | M | R | 114 | 64 | 52 | 18 | 21.8 | 0.717 | 225 |
| 3 | 62 | M | L | 134 | 67 | 53 | 16 | 23 | 0.76 | 321 |
| 3 | 68 | F | L | 106 | 48 | 46 | 18 | 24.5 | 0.693 | 224 |
| 3 | 43 | M | L | 136 | 65 | 57 | 22 | 25.6 | 0.731 | 351 |
| 3 | 35 | F | R | 124 | 41 | 67 | 20 | 26.3 | 0.692 | 286 |
| 3 | 23 | F | R | 127 | 78 | 62 | 17 | 39.2 | 0.714 | 349 |
| 3 | 52 | M | R | 125 | 59 | 44 | 27 | 39.3 | 0.703 | 545 |
| 3 | 67 | F | R | 114 | 55 | 36 | 9 | 41.5 | 0.73 | 310 |

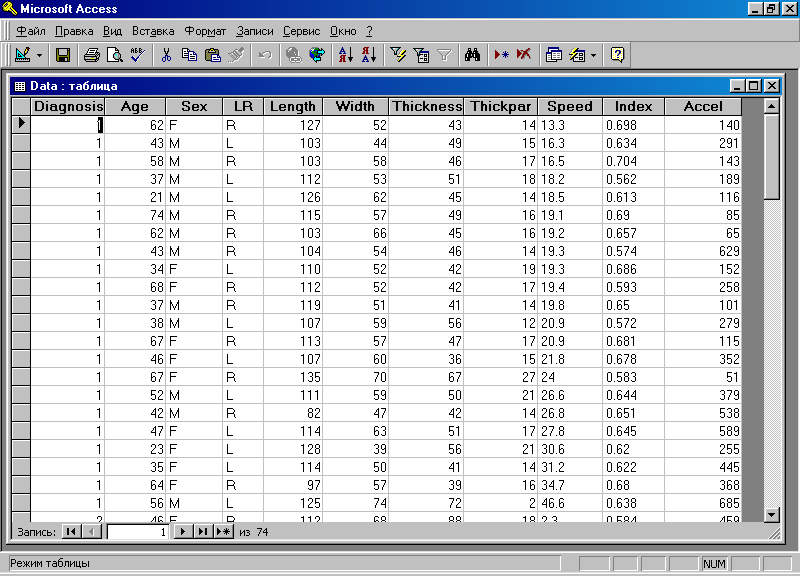
Описание данных эксперимента приведено в таблице№2.

###### *Таблица №2*

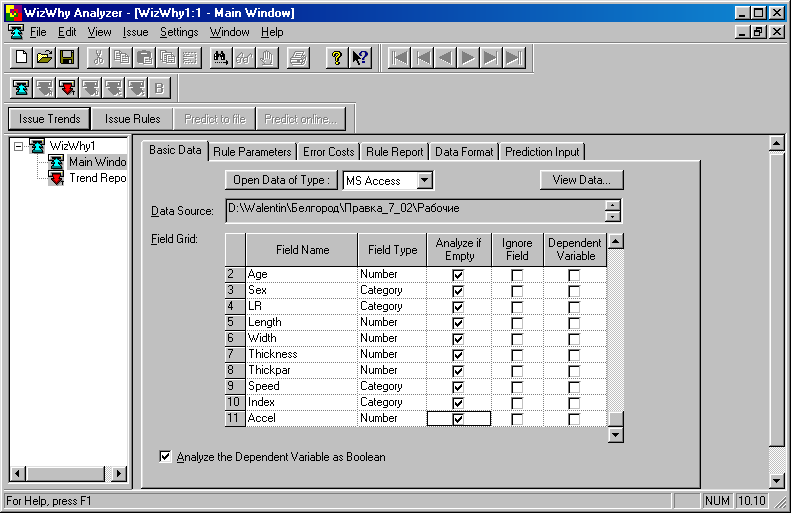
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Diagnosis | Age | Sex | LR | Length | Width | Thickness | Thickpar | Speed | Index | Accel |
| Вариант диагноза | Воз-раст | Пол | Пра-вая или ле-вая почка | Рост | Вес | Параметр зондиро-вания | Параметр зондирования | Крово-ток | Индекс рези-стентности | Пара-метр зонди-рования |

Так как система WIZWHY поддерживает форматы баз данных, введем

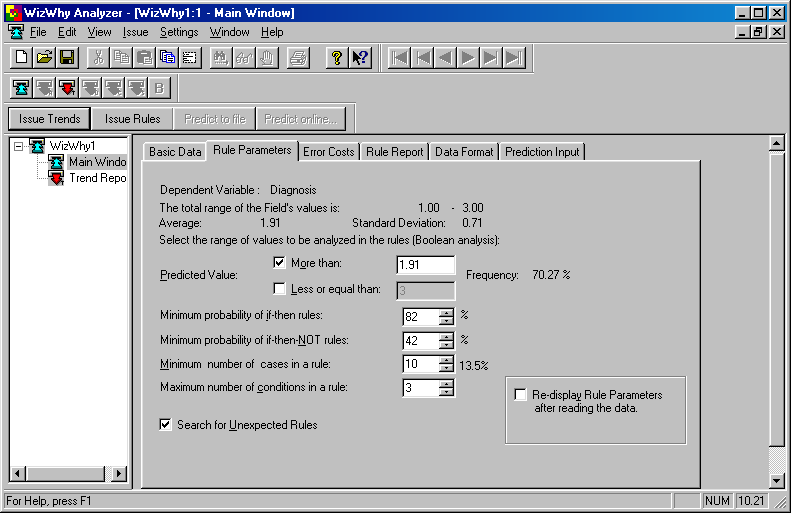
данные таблицы №1 в таблицу СУБД ACCESS.



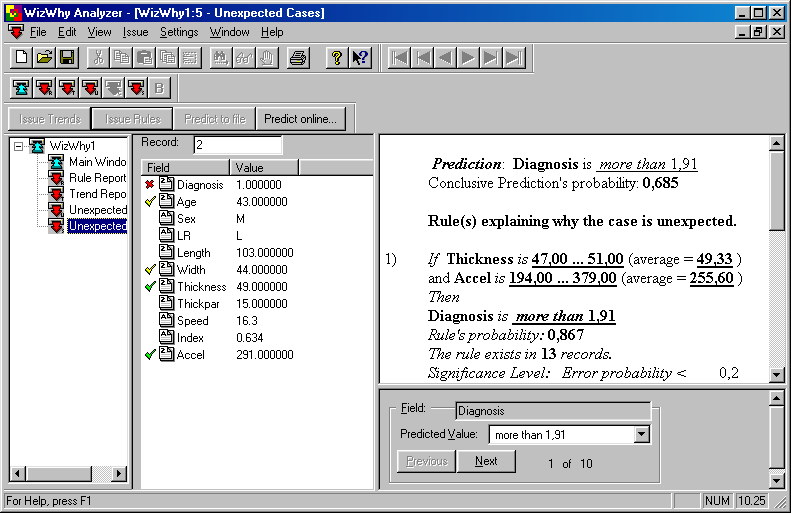
2. В поле **Open Data of Type** укажите тип загружаемых данных и путь к данным.



3. Задание параметров процедуры поиска правил **Rule Parameters:**



1. После того, как все параметры заданы необходимо нажать на кнопку **Issue Rules.**



1. Проанализируйте данные, полученные в результате моделирования и сделайте выводы по работе.
2. Создайте свой файл данных по аналогии с данными, приведенными в таб.№1. Получите продукционную модель и проанализируйте ее.

# Лабораторная работа №3

## **3.1 Цель работы**

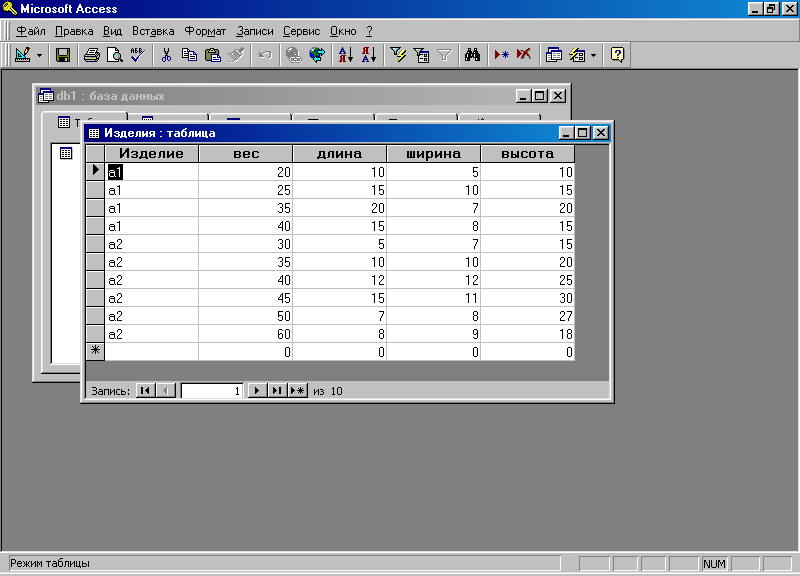
Построение модели предметной области (выявление всех IF THEN правил) и анализ вероятностных характеристик этих моделей.

Целью лабораторной работы является:

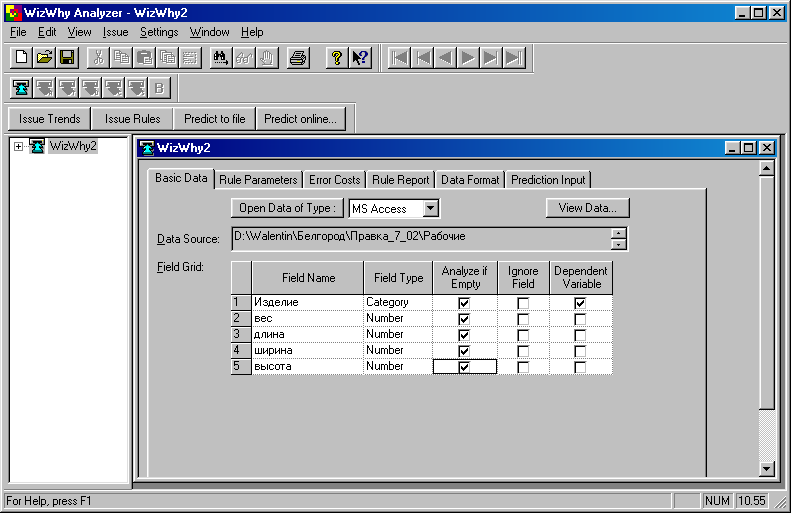
* Построить на первом этапе продукционную модель предметной области типа IF THEN.
* Классифицировать на втором этапе принадлежность новых данных (не входящих в обучающую выборку) к одному из классов.

## **3.2 Порядок выполнения лабораторной работы**

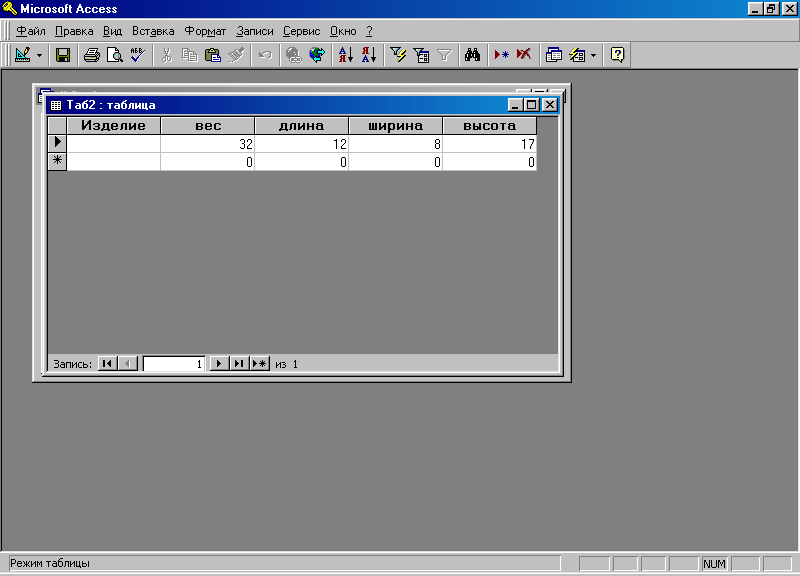
1. При помощи СУБД ACCESS создать таблицу следующей структуры с именем «Модель»:



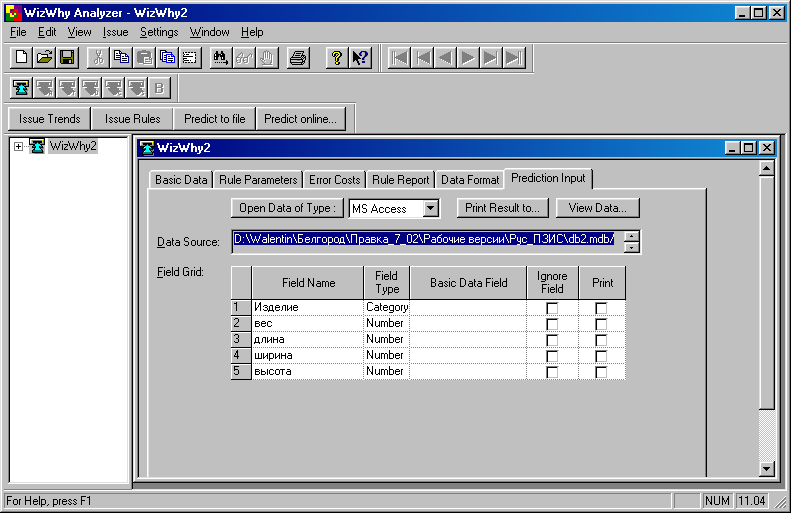
2. В поле **Open Data of Type** укажите тип загружаемых данных и путь к данным.



3. При помощи СУБД ACCESS создать таблицу с именем «Классификация», ввести одну строку данных, которые система должна классифицировать, т.е. с определенной вероятностью отнести к тому или иному классу.



4. Произведем настройки закладки **Prediction Input**



5. После того как все параметры заданы, необходимо нажать на кнопку **Issue Rules,** система построит модель и определит принадлежность входных данных к одному из классов.

6. Сделайте выводы по работе.

# Лабораторная работа №4

## **4.1 Цель работы**

Построение фрагмента экспертной системы. Анализ данных находящихся в таблицах СУБД ACCESS.

## **4.2 Порядок выполнения работы**

1. Выберите предметную область для построения экспертной системы.
2. Постройте таблицу зависимостей (АТРИБУТ ОБЪЕКТ ЗНАЧЕНИЕ).
3. При помощи системы WIZWHY постройте продукционную модель.
4. Произведите настройки системы на решение задачи идентификации входных данных одному из классов.
5. Сделайте отчет по лабораторной работе.

# Список рекомендуемой литературы

1. И. Братко. Программирование на языке Пролог для искусственного интеллекта.- М.: Мир, 1990.

2. Г. Долин. Что такое ЭС.- Компьютер Пресс, 1992/2.

3. Д. Р. Малпасс. Реляционный язык Пролог и его применение.

4. Д. Н. Марселлус. Программирование экспертных систем на Турбо Прологе.- М.: Финансы и статистика, 1994.

5. К. Нейлор. Как построить свою экспертную систему.- М.: Энергоатомиздат, 1991.

6. Н. Д. Нильсон. Искусственный интеллект. Методы поиска решений.- М.: Мир, 1973.

7. В. О. Сафонов. Экспертные системы- интеллектуальные помощники специалистов.- С.-Пб: Санкт-Петербургская организация общества “Знания” России, 1992.

8. К. Таунсенд, Д. Фохт. Проектирование и программная реализация экспертных систем на персональных ЭВМ.- М.: Финансы и статистика, 1990.

9. В. Н. Убейко. Экспертные системы.- М.: МАИ, 1992.

10. Д. Уотермен. Руководство по экспертным системам.- М.: Мир, 1980.

11. Д. Элти, М. Кумбс. Экспертные системы: концепции и примеры.- М.: Финансы и статистика, 1987.

**Учебное издание**

#### *Филатов Валентин Александрович*

Представление знаний

в информационных системах

методические указания

для выполнения лабораторных работ

***Технический редактор:* Иванова Н.И.**

***Компьютерный набор:* Филатов В.А.**

***Корректор:* Иванова Н.И.**



Подписано в печать\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бумага для множительной техники

Формат \_\_\_\_\_\_\_ Усл. печ. л.\_\_\_\_\_\_\_\_Тираж \_\_\_\_\_ экз. Заказ \_\_\_\_\_\_\_



**Отпечатано с авторского оригинала**

**в отделе оперативной печати СТИ МИСиС**

**Старый Оскол, м-н Макаренко, 40**