МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ» (ДГТУ)

Авиационно-технологический колледж

УТВЕРЖДАЮ
Директор колледжа
_____ В.А. Зибров

«___»_____2022г

Методические указания по профессиональному модулю

ПМ. 11. Разработка, администрирование и защита баз данных

специальности 09.02.07 Информационные системы и программирование

Ростов-на-Дону.
2022

1 Методические указания по профессиональному модулю ПМ. 02. Осуществление интеграции программных модулей

Профессиональный модуль ПМ. 11. Разработка, администрирование и защита баз данных изучается на втором и третьем курсах во втором и третьем семестрах. В процессе изучения профессионального модуля используются различные виды занятий: лекции, лабораторные и самостоятельные (индивидуальные) занятия, курсовое проектирование. На первом занятии по данному модулю необходимо ознакомить обучающихся с требованиями к его изучению.

В процессе проведения занятий используются следующие образовательные технологии:

- развивающее обучение;
- проблемное обучение;
- информационно-коммуникационные технологии и т.д.

Профессиональный модуль ПМ. 11. Разработка, администрирование и защита баз данных включает в себя:

- МДК.11.01 Технология разработки и защиты баз данных
- УП.11 Учебная практика;
- ПП.11 Производственная практика.

2 Методические рекомендации при работе над конспектом лекций во время проведения теоретических занятий

В ходе учебных занятий необходимо вести конспектирование учебного материала. Обращать внимание на категории, формулировки, раскрывающие содержание тех или иных явлений и процессов, научные выводы и практические рекомендации, положительный опыт в изучении проблем логики. Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал, а также подчеркивающие особую важность тех или иных теоретических положений. Задавать преподавателю уточняющие вопросы с целью уяснения теоретического материала, разрешения спорных ситуаций.

3 Методические рекомендации при подготовке к лабораторным занятиям

Составной частью учебного процесса в колледже являются лабораторные занятия.

Лабораторное занятие - это занятие, проводимое под руководством преподавателя в учебной аудитории, направленное на углубление теоретических знаний и овладение практическим опытом. Перед лабораторным занятием следует изучить теоретический материал, обращая внимание на практическое их применение.

На лабораторном занятии главное уяснить связь решаемых ситуаций с теоретическими положениями. Для ведения записей на лабораторных занятиях обычно ведется журнал лабораторных работ. Логическая связь теоретических и лабораторных занятий заключается в том, что информация, полученная на лекции, осмысливается и перерабатывается, при помощи преподавателя анализируется до мельчайших подробностей, после чего прочно усваивается.

Успешное освоение профессионального модуля требует регулярных, последовательных и систематических занятий.

Выполнение обучающимися лабораторные работ направлено на:

- обобщение, систематизация, углубление, закрепление полученных теоретических знаний по конкретным темам междисциплинарных курсов профессионального модуля

- формирование умений применять полученные знания на практике, реализация единства интеллектуальной и практической деятельности;

- развитие личностных качеств, направленных на устойчивое стремление к самосовершенствованию: самопознанию, самоконтролю, самооценке, саморазвитию и саморегуляции;

- выработку таких профессионально значимых качеств, как самостоятельность, ответственность, точность, творческая инициатива.

Лабораторные занятия включают следующие необходимые структурные элементы:

- инструктаж, проводимый преподавателем;

- самостоятельная деятельность обучающихся;

- обсуждение итогов выполнения лабораторной работы (здания).

Перед выполнением практического задания проводится проверка знаний обучающихся – их теоретической готовности к выполнению задания. Как правило, лабораторные занятия проводятся по темам, по которым ранее давался лекционный материал.

Количество, объем и содержание лабораторные занятий определяются рабочей программой по междисциплинарным курсам профессионального модуля.

Лабораторные занятия по междисциплинарным курсам профессионального модуля направлены на формирование у обучающихся практических и профессиональных умений при решении задач и при выполнении определенных заданий, необходимых в последующей профессиональной деятельности.

Наряду с формированием умений и овладением практического опыта в процессе лабораторных занятий теоретические знания обобщаются, систематизируются, углубляются и конкретизируются.

Содержание лабораторных занятий фиксируется в рабочей учебной программе междисциплинарных курсов профессионального модуля в разделе «Содержание учебной дисциплины» и планируется с расчетом, чтобы за отведенное время они могли быть выполнены большинством обучающихся.

При выполнении заданий обучающиеся имеют возможность пользоваться лекционным материалом, с разрешения преподавателя, осуществлять деловое общение с товарищами.

Оценка компетентности осуществляется следующим образом: по окончании выполнения задания обучающиеся оформляют журнал лабораторных работ, который затем выносится на завершающий этап формы изучения дисциплины.

Методические указания к выполнению лабораторно-практических работ представлены в приложении.

4 Методические рекомендации по выполнению курсовой работы

Курсовая работа, предусмотренная рабочим учебным планом, является важным этапом в усвоении обучающимися изучаемого профессионального модуля. Выполнение курсовой работы рассматривается как вид учебной работы по междисциплинарному курсу МДК.11.01 Технология разработки и защиты баз данных профессионального модуля ПМ.11. и реализуется в пределах времени, отведенного на его изучение. Процесс выполнения курсовой работы способствует формированию у обучающихся профессиональных и общих компетенций, аналитического мышления. В ходе работы над выполнением курсовой работы обучающийся учится грамотно и четко излагать мысли, что важно для выполнения им выпускной квалификационной работы.

Выполнение курсовой работы осуществляется под руководством преподавателя профессионального модуля. Результатом данной работы должен стать курсовая работа,

выполненная и оформленная в соответствии с установленными требованиями. Курсовая работа подлежит обязательной защите. Методические указания к выполнению курсовой работы по МДК.11.01 Технология разработки и защиты баз данных представлены в виде отдельной методической разработки.

5 Методические рекомендации для самостоятельной работы

Самостоятельная работа - это планируемая работа обучающихся, выполняемая по заданию и при методическом руководстве преподавателя, но без его непосредственного участия. Междисциплинарный курс профессионального модуля предусматривает два вида самостоятельной работы:

- аудиторная;
- внеаудиторная.

Самостоятельная работа выполняет ряд функций, среди которых особенно выделяются:

- развивающая (повышение культуры умственного труда, приобщение к творческим видам деятельности, обогащение интеллектуальных способностей обучающихся);
- ориентирующая и стимулирующая (процессу обучения придается ускорение и мотивация);
- воспитательная (формируются и развиваются профессиональные качества специалиста);
- исследовательская (новый уровень профессионально-творческого мышления);
- информационно-обучающая (учебная деятельность обучающихся на аудиторных занятиях).

Целью самостоятельных занятий является самостоятельное более глубокое изучение обучающимися вопросов междисциплинарного курса с использованием рекомендуемой литературы и других информационных источников.

Задачами самостоятельной работы являются:

- систематизация и закрепление полученных теоретических знаний и практических умений обучающихся;
- углубление и расширение теоретических знаний;
- формирование умения использовать справочную литературу;
- развитие познавательных способностей и активности обучающихся: творческой инициативы, ответственности и организованности.

Внеаудиторная самостоятельная работа включает такие формы работы, как:

1) индивидуальные занятия (домашние занятия):

- изучение программного материала дисциплины (работа с учебником и конспектом лекции);
- изучение рекомендуемых литературных источников;
- конспектирование источников;
- работа с нормативными документами;
- работа с электронными информационными ресурсами и ресурсами Internet;
- составление схем, таблиц, для систематизации учебного материала;
- подготовка презентаций
- ответы на контрольные вопросы;
- написание рефератов;

2) групповая самостоятельная работа студентов:

- подготовка к занятиям, проводимым с использованием активных форм обучения (круглые столы, деловые игры и др.);
- анализ деловых ситуаций (мини-кейсов) и др.

3) получение консультаций для разъяснений по вопросам изучаемой дисциплины.

6 Методические рекомендации по учебной и производственной практикам

Практика обучающихся является составной частью образовательного процесса и составной частью основной профессиональной образовательной программы (ОПОП) среднего профессионального образования (СПО), обеспечивающей реализацию Федерального государственного образовательного стандарта (ФГОС) СПО.

Практика имеет целью комплексного освоения обучающимися всех видов профессиональной деятельности по специальности среднего профессионального образования, формирование общих и профессиональных компетенций, а также приобретение необходимых умений и опыта практической работы по специальности 09.02.07 Информационные системы и программирование.

Видами практики обучающихся, осваивающих основную профессиональную образовательную программу (ОПОП) среднего профессионального образования (СПО), являются: учебная практика и производственная практика по профилю специальности.

Основной целью учебной практики является получение первичных профессиональных умений и навыков, формирование общих и профессиональных компетенций.

Производственная практика (по профилю специальности) направлена на формирование у обучающегося общих и профессиональных компетенций, приобретение практического опыта и реализуется в рамках профессиональных модулей ОПОП СПО по каждому из видов профессиональной деятельности, предусмотренных ФГОС СПО по специальности 09.02.07 .

Методические указания по выполнению учебной практики УП.11 и производственной практик ПП.11 представлены в отдельных методических материалах.

7 Методические рекомендации по работе с литературой

Авиационный колледж обеспечивает учебно-методическую и материально - техническую базу для организации самостоятельной работы обучающихся.

Библиотека обеспечивает:

- учебный процесс необходимой литературой и информацией (комплектует библиотечный фонд учебной, методической, научной, справочной литературой в соответствии с учебными планами и программами, в том числе на электронных носителях);
- доступ к основным информационным образовательным ресурсам, информационной базе данных, в том числе возможность выхода в Интернет.

Колледж:

- обеспечивает доступность всего необходимого учебно-методического и справочного материала.
- разрабатывает: учебно-методические комплексы, программы, пособия, материалы по учебным дисциплинам в соответствии с ФГОС; методические указания по

дисциплинам, методические рекомендации по планированию и организации самостоятельной работы обучающихся, методические рекомендации по выполнению ВКР и т.д

8 Описание мультимедийных средств

В преподавании дисциплины новые возможности, предоставляемые мультимедийными средствами, нашли самое разнообразное применение.

На занятиях используются несложные мультимедийные документы, которые может создать сам преподаватель и обучающийся в программах Microsoft Word или Power Point и т.д.

Системный подход в мультимедийных проектах даёт возможность использовать средства визуальной информации не только в качестве иллюстративного материала, сопровождающего рассказ преподавателя, но и структурировать подготовленные материалы применительно к специализациям обучающихся. Все это положительно влияет на усвоение материалов и поднимает уровень и качество обученности.

Приложение

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ ЛАБОРАТОРНО-ПРАКТИЧЕСКИХ РАБОТ
МДК.11.01 Технология разработки и защиты баз данных

Методические указания к выполнению
практического занятия № 1

Тема: Построение концептуальной модели базы данных

Цель: приобретение практических навыков анализа предметной области, информационных задач и построения концептуальной модели базы данных

Выполнение работы

1. Теоретическое обоснование

Проектирование базы данных (БД) – одна из наиболее сложных и ответственных задач, связанных с созданием информационной системы (ИС). В результате её решения должны быть определены содержание БД, эффективный для всех её будущих пользователей способ организации данных и инструментальные средства управления данными.

Основная цель процесса проектирования БД состоит в получении такого проекта, который удовлетворяет следующим требованиям:

- корректность схемы БД, т.е. база должна быть гомоморфным образом моделируемой предметной области (ПО), где каждому объекту предметной области соответствуют данные в памяти ЭВМ, а каждому процессу – адекватные процедуры обработки данных;
- обеспечение ограничений ;
- эффективность функционирования ;
- защита данных (от аппаратных и программных сбоев и несанкционированного доступа);
- простота и удобство эксплуатации;
- гибкость, т.е. возможность развития и адаптации к изменениям предметной области и/или требований пользователей.

Базы данных всегда проектируются под конкретное назначение системы.

Этапы проектирования базы данных

Процесс проектирования включает в себя следующие этапы:

Концептуальное проектирование – это процедура конструирования информационной модели, не зависящей от каких-либо физических условий реализации.

Логическое проектирование – это процесс конструирования информационной модели на основе существующих моделей данных, не зависимо от используемой СУБД и других условий физической реализации.

Физическое проектирование – это процедура создания описания конкретной реализации БД с описанием структуры хранения данных, методов доступа к данным.

Концептуальное проектирование

Основными задачами концептуального проектирования являются определение предметной области системы и формирование взгляда на ПО с позиций сообщества будущих пользователей БД, т.е. инфологической модели ПО.

Концептуальная модель ПО представляет собой описание структуры и динамики ПО, характера информационных потребностей пользователей в терминах, понятных пользователю и не зависимых от реализации БД. Это описание выражается в терминах не отдельных объектов ПО и связей между ними, а их типов, связанных с ними ограничений целостности и тех процессов, которые приводят к переходу предметной области из одного состояния в другое.

Проектирование с использованием метода "сущность-связь"

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

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

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

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

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

Для каждой сущности выбираются свойства (атрибуты). Различают:

- Идентифицирующие и описательные атрибуты. Идентифицирующие атрибуты имеют уникальное значение для сущностей данного типа и являются потенциальными ключами. Они позволяют однозначно распознавать экземпляры сущности. Из потенциальных ключей выбирается один первичный ключ (ПК). В качестве ПК обычно выбирается потенциальный ключ, по которому чаще происходит обращение к экземплярам записи. Кроме того, ПК должен включать в свой состав минимально необходимое для идентификации количество атрибутов. Остальные атрибуты называются описательными и заключают в себе интересующие свойства сущности.
- Составные и простые атрибуты. Простой атрибут состоит из одного компонента, его значение неделимо. Составной атрибут является комбинацией нескольких компонентов, возможно, принадлежащих разным типам данных (например, ФИО или адрес). Решение о том, использовать составной атрибут или разбивать его на компоненты, зависит от характера его обработки и формата пользовательского представления этого атрибута.

- Однозначные и многозначные атрибуты (могут иметь соответственно одно или много значений для каждого экземпляра сущности).
- Основные и производные атрибуты. Значение основного атрибута не зависит от других атрибутов. Значение производного атрибута вычисляется на основе значений других атрибутов (например, возраст студента вычисляется на основе даты его рождения и текущей даты).

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

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

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

По типу различают множественные связи "один к одному" (1:1), "один ко многим" (1:N) и "многие ко многим" (M:N).

Степень связи определяется количеством сущностей, которые охвачены данной связью. Пример бинарной связи – связь между отделом и сотрудниками, которые в нём работают. Примером тернарной связи является связь типа экзамен между сущностями ДИСЦИПЛИНА, СТУДЕНТ, ПРЕПОДАВАТЕЛЬ. Из последнего примера видно, что связь также может иметь атрибуты (в данном случае это Дата проведения и Оценка). Пример ER–диаграммы с указанием сущностей, их атрибутов и связей приведен на рис. 1.

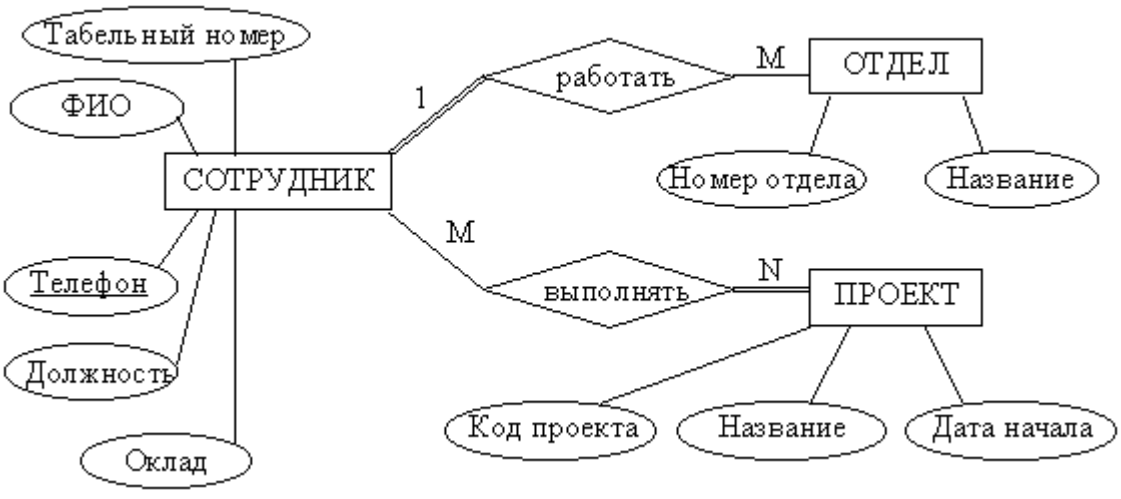


Рисунок 1 - Пример ER–диаграммы с однозначными и многозначными атрибутами

Анализ предметной области

В качестве примера возьмем базу данных компании, которая занимается издательской деятельностью.
База данных создаётся для информационного обслуживания редакторов, менеджеров и других сотрудников компании. БД должна содержать данные о сотрудниках компании, книгах, авторах, финансовом состоянии компании и предоставлять возможность получать разнообразные отчёты.
В соответствии с предметной областью система строится с учётом следующих особенностей:

- каждая книга издаётся в рамках контракта;

- книга может быть написана несколькими авторами;
- контракт подписывается одним менеджером и всеми авторами книги;
- каждый автор может написать несколько книг (по разным контрактам);
- порядок, в котором авторы указаны на обложке, влияет на размер гонорара;
- если сотрудник является редактором, то он может работать одновременно над несколькими книгами;
- у каждой книги может быть несколько редакторов, один из них – ответственный редактор;
- каждый заказ оформляется на одного заказчика;
- в заказе на покупку может быть перечислено несколько книг.

Выделим базовые сущности этой предметной области:

1. Сотрудники компании. Атрибуты сотрудников – ФИО, табельный номер, пол, дата рождения, паспортные данные, ИНН, должность, оклад, домашний адрес и телефоны. Для редакторов необходимо хранить сведения о редактируемых книгах; для менеджеров – сведения о подписанных контрактах.
2. Авторы. Атрибуты авторов – ФИО, ИНН (индивидуальный номер налогоплательщика), паспортные данные, домашний адрес, телефоны. Для авторов необходимо хранить сведения о написанных книгах.
3. Книги. Атрибуты книги – авторы, название, тираж, дата выхода, цена одного экземпляра, общие затраты на издание, авторский гонорар.
4. Контракты будем рассматривать как связь между авторами, книгами и менеджерами. Атрибуты контракта – номер, дата подписания и участники.
5. Для отражения финансового положения компании в системе нужно учитывать заказы на книги. Для заказа необходимо хранить номер заказа, заказчика, адрес заказчика, дату поступления заказа, дату его выполнения, список заказанных книг с указанием количества экземпляров.

ER–диаграмма издательской компании приведена на рис. 2 (базовые сущности на рисунках выделены полужирным шрифтом).

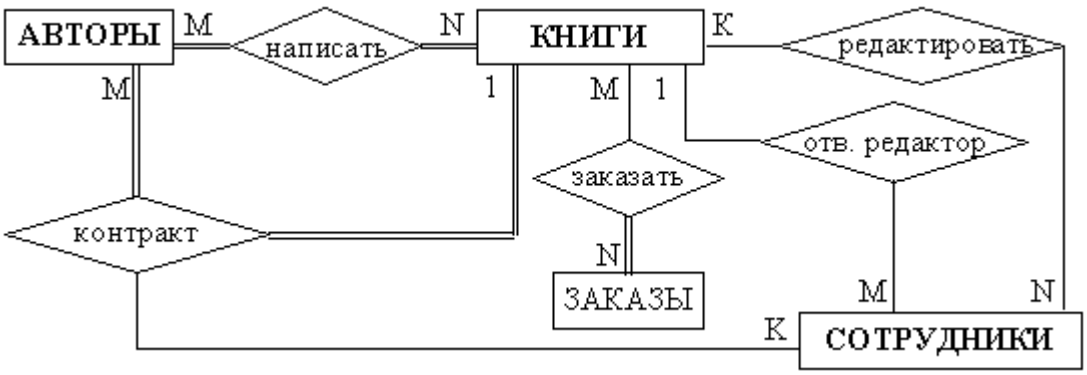


Рисунок 2 - ER–диаграмма издательской компании

2. Алгоритм выполнения работы

Анализ информационных задач и круга пользователей системы

Система создаётся для обслуживания следующих групп пользователей:

- администрация (дирекция);
- менеджеры;
- редакторы;
- сотрудники компании, обслуживающие заказы.

Определим границы информационной поддержки пользователей:

1) Функциональные возможности:

- ведение БД (запись, чтение, модификация, удаление в архив);
- обеспечение логической непротиворечивости БД;
- обеспечение защиты данных от несанкционированного или случайного доступа (определение прав доступа);
- реализация наиболее часто встречающихся запросов в готовом виде;
- предоставление возможности сформировать произвольный запрос на языке манипулирования данными.

2) Готовые запросы:

- получение списка всех текущих проектов (книг, находящихся в печати и в продаже);
- получение списка редакторов, работающих над книгами;

- получение полной информации о книге (проекте);
- получение сведений о конкретном авторе (с перечнем всех книг);
- получение информации о продажах (по одному или по всем проектам);
- определение общей прибыли от продаж по текущим проектам;
- определение размера гонорара автора по конкретному проекту.

Задание по работе: по заданному описанию предметной области построить концептуальную модель базы данных :

- Выделите типы сущностей;
- Выделите типы связей и определите для них показатели кардинальности и степень участия сторон;
- Выделите атрибуты и свяжите их типами сущностей и связей;
- Определите потенциальные и первичные ключи сущностей;
- Нарисуйте ER-диаграмму.

и проанализируйте информационные задачи и группы пользователей

Контрольные вопросы.

1. Назовите основные этапы проектирования базы данных.
2. Какова основная цель этапа концептуального проектирования.
3. Объясните смысл понятия «представление пользователя » и укажите источники информации, которые могут использоваться при создании этого представления

Методические указания к выполнению практического занятия № 2-3

Тема: Инфологическая модель базы данных

Цель: приобретение практических навыков анализа предметной области, информационных задач и построения инфологической модели базы данных

1.Теоретическое обоснование

Инфологическая модель применяется на втором этапе проектирования БД. Инфологическое проектирование прежде всего связано с попыткой представления семантики, то есть смыслового содержания, предметной области в модели БД.

Реляционная модель данных в силу своей простоты и лаконичности не позволяет отобразить семантику, то есть смысл предметной области остается за рамками реляционной модели. Ранние теоретико-графовые модели в большей степени, чем реляционная модель, отображали семантику предметной области. Они в явном виде определяли иерархические связи между объектами предметной области.

2. Алгоритм выполнения работы

Построить инфологическую модель по заданному индивидуальному описанию предметной области построить концептуальную модель базы данных.

Вариант1.

Задача – организация учебного процесса в вузе:

* Студенты: паспортные данные, адрес, дата зачисления, номер приказа, факультет, группа, является ли старостой, кафедра (специализация), изучаемые (изученные) предметы, оценки, задолженности, стипендия.

* Учебные курсы: название, факультет(ы), групп(ы), кафедра, семестр(ы), форма отчётности, число часов.

* Преподаватели: паспортные данные, адрес, телефон, фотография, кафедра, должность, учёная степень, начальник (зав. кафедрой), предмет(ы), число ставок, зарплата.

.Вариант2.

Учет и выдача книг в библиотеке вуза:

* Книги: авторы, название, раздел УДК, раздел (техническая, общественно-политическая и т.п.), место и год издания, издательство, количество страниц, иллюстрированность, цена, дата покупки, номер сопроводительного документа (чек, счёт/накладная), вид издания (книги, учебники, брошюры, периодические издания), инвентарный номер (есть только для книг и некоторых учебников), длительность использования читателями (год, две недели, день), электронная версия книги или ее реферата (отсканированный текст).

* Читатели: номер читательского билета, ФИО, год рождения, адрес, дата записи, вид

(студент, аспирант, преподаватель, сотрудник), курс, номер группы, названия взятых книг и даты их выдачи.

Вариант3.

Отдел кадров некоторой компании.

* Сотрудники: ФИО, паспортные данные, фотография, дом. и моб. телефоны, отдел, комната, раб. телефоны (в т.ч. местный), подчинённые сотрудники, должность, тип(ы) работы, задание(я), проект(ы), размер зарплаты, форма зарплаты (почасовая, фиксированная).
* Отделы: название, комната, телефон(ы), начальник, размер финансирования, число сотрудников.
* Проекты: название, дата начала, дата окончания, размер финансирования, тип финансирования (периодический, разовый), задачи и их исполнители, структура затрат и статьи расходов.

Вариант4.

Отдел поставок некоторого предприятия:

* Поставщики: название компании, ФИО контактного лица, расчётный счёт в банке, телефон, факс, поставляемое оборудование (материалы), даты поставок (по договорам и реальные), метод и стоимость доставки.
* Сырьё: тип, марка, минимальный запас на складе, время задержки, цена, продукты, при производстве которых используется, потребляемые объемы (необходимый, реальный, на единицу продукции).

Вариант5.

Пункт проката видеозаписей (внутренний учет).

* Видеокассеты: идентификационный номер видеокассеты, тип видеокассет, дата его создания, компания-поставщик, число штук данного типа (общее, в магазине, выдано в настоящее время, выдано всего, выдано в среднем за месяц), общая длительность записей; записи видеокассет: название, длительность, категория, год выпуска и производитель (оригинала).
* Клиенты: ФИО, паспортные данные, адрес, телефон; заказы, т.е. взятые видеокассеты (сейчас и в прошлом): номер, дата выдачи, дата возвращения, общая стоимость заказа.

Вариант6.

Пункт проката видеозаписей (информация для клиентов).

* Видеокассеты: краткое описание, внешний вид (этикетка), марка (пустой) видеокассеты, цена за единицу прокатного времени (например: 1 день, 3 дня, неделя), есть ли в наличии, общая длительность записей; записи на видеокассете: название, длительность, жанр (категория), тема, год и страна выпуска (оригинала), кинокомпания, описание, актеры, режиссер.
* Заказы: идентификационные номера и названия выданных видеокассет, дата выдачи, дата возвращения (продления), общая стоимость заказа, возвращены ли кассеты заказа.

Вариант7.

Кинотеатры (информация для зрителей).

* Фильмы: название, описание, жанр (категория), длительность, популярность (рейтинг, число проданных билетов в России и в мире), показывается ли сейчас (сегодня, на текущей неделе), в каких кинотеатрах показывается, цены на билеты (в т.ч. средние).
* Кинотеатры: название, адрес, схема проезда, описание, число мест (в разных залах, если их несколько), акустическая система, широкоэкранность, фильмы и цены на них: детские и взрослые билеты в зависимости от сеанса (дневной, вечерний и т.п.) и от категории мест (передние, задние и т.п.); сеансы показа фильмов (дата и время начала).

Вариант8.

Ресторан (информация для посетителей).

* Меню: дневное или вечернее, список блюд по категориям.

* Блюда: цена, название, вид кухни, категории (первое, второе и т.п.; мясное, рыбное, салат и т.п.), является ли вегетарианским, компоненты блюда, время приготовления, есть ли в наличии.
* Компоненты блюд: тип (гарнир, соус, мясо и т.п.), калорийность, цена, рецепт, время приготовления, есть ли в наличии, ингредиенты (продукты) и их расходы на порцию.

Вариант9.

Задача- информационная поддержка деятельности склада.

База данных должна содержать информацию о наименовании товара, его поставщике, количестве, цене товара, конечном сроке реализации, сроке хранения на складе. Торговый склад производит уценку хранящейся продукции. Если продукция хранится на складе дольше 10 месяцев, то она уценивается в 2 раза, а если срок хранения превысил 6 месяцев, но не достиг 10, то в 1,5 раза. Ведомость уценки товаров должна содержать информацию: наименование товара, количество товара(шт.), цена товара до уценки, срок хранения товара, цена товара после уценки, общая стоимость товаров после уценки.

Вариант10.

Задача – информационная поддержка деятельности адвокатской конторы. БД должна осуществлять:

ведение списка адвокатов;

ведение списка клиентов;

ведение архива законченных дел.

Необходимо предусмотреть:

получение списка текущих клиентов для конкретного адвоката;

определение эффективности защиты (максимальный срок минус полученный срок) с учётом оправданий, условных сроков и штрафов;

определение неэффективности защиты (полученный срок минус минимальный срок);

подсчёт суммы гонораров (по отдельных делам) в текущем году;

получение для конкретного адвоката списка текущих клиентов, которых он защищал ранее (из архива, с указанием полученных сроков и статей).

Вариант11.

Задача – информационная поддержка деятельности гостиницы.

БД должна осуществлять:

ведение списка постояльцев;

учёт забронированных мест;

ведение архива выбывших постояльцев за последний год.

Необходимо предусмотреть:

получение списка свободных номеров (по количеству мест и классу);

получение списка номеров (мест), освобождающихся сегодня и завтра;

выдачу информации по конкретному номеру;

автоматизацию выдачи счетов на оплату номера и услуг;

получение списка забронированных номеров;

проверку наличия брони по имени клиента и/или названию организации

Вариант12.

Описание предметной области:

- В компании несколько отделов.
- В каждом отделе есть некоторое количество сотрудников, занятых в нескольких проектах и размещающихся в нескольких офисах.
- Каждый сотрудник имеет план работы, т.е. несколько заданий, которые он должен выполнить. Для каждого такого задания существует ведомость, содержащая перечень денежных сумм, полученных сотрудником за выполнение этого задания.
- В каждом офисе установлено несколько телефонов.

В базе данных должна храниться следующая информация.

- Для каждого отдела: номер отдела (уникальный), его бюджет и личный номер сотрудника, возглавляющего отдел (уникальный).
- Для каждого сотрудника: личный номер сотрудника (уникальный), номер текущего проекта, номер офиса, номер телефона, название выполняемого задания вместе с датой и размером выплат, проведенных в качестве оплаты за выполнение данного задания.
- Для каждого проекта : номер проекта (уникальный) и его бюджет.
- Для каждого офиса : номер офиса (уникальный), площадь в квадратных футах, номера всех установленных в нем телефонов.

Вариант13.
Задача – информационная поддержка деятельности спортивного клуба. БД должна осуществлять:
ведение списков спортсменов и тренеров;
учёт проводимых соревнований (с ведением их архива);
учёт травм, полученных спортсменами.
Необходимо предусмотреть:
возможность перехода спортсмена от одного тренера к другому;
составление рейтингов спортсменов;
составление рейтингов тренеров;
выдачу информации по соревнованиям;
выдачу информации по конкретному спортсмену;
подбор возможных кандидатур на участие в соревнованиях (соответствующего уровня мастерства, возраста и без травм).
Вариант14.
Задача – информационная поддержка деятельности аптечного склада.
В аптечном складе хранятся лекарства. Сведения о лекарствах содержатся в специальной ведомости: наименование лекарственного препарата; количество (в шт.); цена; срок хранения на складе (в месяцах). Лекарства поступают на склад ежедневно от разных поставщиков, отпускаются два раза в неделю по предварительным заказам аптек. Выяснить, сколько стоит самый дорогой и самый дешевый препарат; сколько препаратов хранится на складе более 3 месяцев; сколько стоят все препараты, хранящиеся на складе, отыскать препараты, остаток которых равен нулю , ниже требуемого по заказам.
Вариант15.
“Электронный журнал посещаемости"
Предметная область представлена следующими документами:
Список студентов
Журнал посещаемости
Расписание занятий
Предусмотреть учет пропусков по уважительным, неуважительным причинам. Подсчет пропусков по каждому студенту, за неделю, месяц, заданный период, по конкретному предмету.
Вариант16.
«Итоги сессии»
База данных должна содержать информацию о двух последних сессиях студентов. Источником информации являются экзаменационные ведомости. Необходимо проводить анализ успеваемости по специальностям, формам обучения, курсам, группам, предметам, вычислять средний балл по указанным критериям, а также число каждых оценок .

Контрольные вопросы.

1. Назовите основные этапы проектирования базы данных.
2. Какова основная цель этапа концептуального проектирования.
3. Объясните смысл понятия «представление пользователя » и укажите источники информации, которые могут использоваться при создании этого представления.

Методические указания к выполнению
практического занятия № 4-5

Тема: Создание запросов с использованием конструктора

Цель: приобретение практических навыков работы в режиме конструктора для создания запросов.

1.Теоретическое обоснование

Запросы в MS Access доступны в трех режимах: табличном, в котором запрос выглядит так же, как обычная таблица, режиме конструктора, где запрос предстает в виде схемы связанных объектов

Для того чтобы создать запрос-выборку, нужно перейти на вкладку "Запросы" в окне базы данных и нажать кнопку "Создать", в результате откроется диалоговое окно.

При создании запроса с помощью *Конструктора* разработчику необходимо выбрать таблицы или запросы, содержащие нужные данные и заполнить бланк запроса. *Бланк запроса* расположен в нижней части окна конструктора и содержит описание запроса в табличной форме (см. рисунок 6.3.2). Этот вариант следует выбирать для создания запроса "с нуля" в режиме конструктора.

Каждая колонка в бланке запроса соответствует одному полю. Строки "Поле" и "Имя таблицы" содержат списки, которые позволяют определить нужное поле. Таблица бланка запросов расширяется вправо автоматически при добавлении в запрос новых полей. Контекстное меню, связанное с бланком запроса, позволяет скрывать или, наоборот, показывает строки "Имя таблицы" и "Групповая операция". Под *"Групповой операцией"* понимается обработка и обобщение значений данного поля при помощи определенной функции (Например функция "Count" позволяет подсчитать число записей).

Если воспользоваться вариантом *Простой запрос*, то откроется диалоговое окно Мастера простых запросов, в котором следует выбрать участвующие в запросе поля из любых таблиц или запросов, входящих в базу данных. В последнем диалоговом окне Мастера пользователю предлагается выбор выполнить запрос или просмотреть его структуру в режиме конструктора. Основное преимущество этого способа заключается в том, что Мастер автоматически заполняет бланк запроса. Если полученный запрос не соответствует требованиям, в него можно внести изменения в режиме конструктора.

Перекрестный запрос - это операция построения таблицы для вычисления итоговых значений на основе существующей таблицы или запроса. Перекрестный запрос создается в предположении, что исходная таблица содержит необходимые данные для формирования заголовков строк и столбцов новой таблицы. Например, из всего объема продаж товаров и услуг за всю историю предприятия необходимо выбрать показатели по конкретным позициям за конкретный год, несколько определенных месяцев или кварталов.

В том случае, когда из всего массива данных одной или нескольких таблиц нужно выбрать лишь те данные, которые повторяются, следует воспользоваться вариантом *"Повторяющиеся записи"*. Например, выбрать те товары, количество продаж которых "больше чем один".

Если нужно найти среди всех записей базы данных те, которые не имеют подчиненных, следует выбирать вариант, "Записи без подчиненных". Например, это достаточно типовая ситуация для любой базы данных, фиксирующей торговые операции по конкретным клиентам или по конкретным товарам. Незачем хранить информацию по конкретной поставке товара, если сам товар уже давно продан. Также незачем хранить в списке "постоянных клиентов" фамилии тех людей, которые уже длительное время не осуществляли никаких покупок. Очевидно, что чем меньше записей приходится просматривать, тем быстрее работает вся СУБД.

Запрос с параметрами - это запрос, при выполнении которого в его диалоговом окне пользователю выдается приглашение ввести данные, например условие для возвращения записей или значение, которое должно содержаться в поле. Можно создать запрос, в результате которого выводится приглашение на ввод нескольких данных, например, двух

дат. В результате будут возвращены все записи, находящиеся между указанными двумя датами. Также запросы с параметрами удобно использовать в качестве основы для форм и отчетов. Например, на основе запроса с параметрами можно создать месячный отчет о доходах. При выводе данного отчета, на экране появится приглашение ввести месяц, доходы которого интересуют пользователя. После ввода месяца на экране будет представлен требуемый отчет. Можно создать специальную форму или диалоговое окно, которое вместо диалогового окна запроса с параметрами будет выводить приглашение на ввод параметров запроса.

2. Алгоритм выполнения работы

Результат выполнения индивидуального задания по построению концептуальной модели в предыдущей работе использовать для создания базы данных в MS Access.

Разработать необходимые запросы, формы и отчеты.

Контрольные вопросы

1) Что такое запрос в БД?
2) Назначение форм и отчетов при работе с БД
3) Каковы особенности создания запросов с использованием конструктора

Методические указания к выполнению практического занятия № 6-8

Тема: Создание запросов на выборку данных с помощью SQL

Цель: приобретение практических навыков создания запросов с помощью SQL.

1.Теоретическое обоснование

Небольшой набор команд SQL и определенный синтаксис составления выражений позволяет совершать сколь угодно сложные операции. Самой распространенной из них является выборка связанных данных из одной или нескольких таблиц при помощи оператора SELECT, однако в случае необходимости SQL может вносить информацию, корректировать хранящиеся записи и создавать новые объекты базы данных. SQL применяется не только в MS Access, но и во всех современных СУБД.

SQL-запросы построены на лексическом диалекте, основанном на семантике обычного языка (английского). При создании языка предполагалось, что конструкцию на нем сможет написать любой неподготовленный пользователь, руководствуясь исключительно своими потребностями и исходя из существующей структуры данных. Среди базовых инструкций, на которых основываются запросы, стоит выделить следующие наиболее распространенные команды:

1. SELECT - предназначается для выборки из базы данных набора записей, соответствующих указанному критерию. В качестве входных параметров инструкции передаются названия полей таблиц, которые должны быть внесены в результирующий набор.
2. UPDATE - служит для редактирования записей (одной или нескольких). Пользователь указывает названия полей таблиц и их новые значения.
3. DELETE - инструкция, полностью удаляющая из базы данных все указанные записи (а не только значения отдельных полей).

CREATE - позволяет создавать новые объекты базы данных

Вспомогательные конструкции языка, например, WHERE задает критерий уточняющего поиска среди отобранных записей.

В языке запросов, реализованном в Microsoft Access, поддерживаются четыре дополнительных расширения: TRANSFORM (позволяет строить перекрестные запросы), IN (создание связи с удаленной базой данных), DISTINCTROW (определение запроса-объединения с потенциальной возможностью объединения данных), WITH OWNER-ACCESS OPTION (позволяет создавать специализированные запросы, которые может выполнять пользователь, не имеющий права доступа к таблицам, которыми оперирует данный запрос). В качестве функции могут быть использованы итоговые функции SQL, например такие, как AVG, COUNT, MAX, MIN, STDEV, STDEVP, SUM, VAR или VARP

2.Алгоритм выполнения работы

Создать запросы для отбора записей из созданной в предыдущей работе базы с использованием условий отбора:

Знак	Пример	Использование
*	ко* ищет "кот", "котелок" и "кофе", *ка ищет "кнопка", "папка" и "практика"	Соответствует любому количеству символов и может быть использован в любом месте текстовой строки
?	п?ник находит "подлинник", "помощник" и "противник"	Соответствует любому одиночному символу
#	5#2 находит 502, 572, 592	Соответствует любой одиночной цифре
[]	л[аи]па находит "лапа" и "липа", но не "лупа"	Соответствует любому одному символу из заключенных в квадратные скобки
!	л[!аи]па находит "лупа", но не "лапа" и не "липа"	Соответствует любому одному символу, кроме заключенных в скобки
-	б[а-в]д находит "бад", "ббд" и "бвд"	Соответствует любому символу из диапазона символов. Необходимо указывать этот диапазон по возрастанию (например от А до Я, но не от Я до А)

Оператор *BETWEEN...AND* определяет принадлежность значения выражения указанному диапазону

Оператор *IN* проверяет, совпадает ли значение выражения с одним из элементов указанного списка, который задается в круглых скобках.

Оператор *AND* требует одновременного выполнения всех выражений, которые он соединяет.

Оператор *OR* требует выполнения хотя бы одного из тех выражений, которые он соединяет.

Оператор *NOT* требует невыполнения того выражения, перед которым он стоит.

Создайте запросы-действия

Запрос SQL - это запрос, создаваемый при помощи инструкций SQL. Примерами запросов SQL могут служить запросы на объединение, запросы к серверу, перекрестные и подчиненные запросы.

Запрос на *объединение* - комбинирует поля из одной или нескольких таблиц или запросов в одно поле в результатах запроса. Например, если шесть поставщиков ежемесячно посылают новые списки оборудования, то с помощью запроса на объединение эти списки можно объединить в один. А затем результаты поместить в новую таблицу, созданную с помощью запроса на создание таблицы, основанного на запросе на объединение.

Запрос *к серверу* - отправляет команды непосредственно в базы данных ODBC, например Microsoft SQL Server, причем используются только команды, поддерживаемые данным сервером.

Контрольные вопросы

1) Базовые инструкции для создания SQL-запросов
2) Для чего нужны перекрестные запросы
3) Что выполняет запрос на объединение

Методические указания к выполнению практического занятия № 9-10

Тема: Создание хранимых процедур триггеров и генераторов

Цель: Изучить используемые триггеры, генераторы и исключения.

1.Теоретическое обоснование

Генераторы используются для генерации уникальных номеров, используемых в ключевых полях (автоинкрементных полях). То есть они реализуют те же функциональные возможности в таблицах Firebird, что и поля с автоматическим приращением в других СУБД. Можно указать, чтобы числа генерировались, начиная с заданного начального значения.

Пример создания генератора:

```
CREATE GENERATOR MYGENERATOR;
SET GENERATOR MYGENERATOR TO 1000;
```

Генераторы непосредственно не привязываются к какому-либо полю. Они просто позволяют генерировать уникальные числа. Для этого используется функция **Gen_ID()**, встроенная в Firebird, которая генерирует целочисленные значения. Она берет генератор в качестве первого параметра и значение шага в качестве второго. Обычно приращение равно 1.

Обращаться к генератору можно только через функцию **Gen_ID()**.

В реальной жизни вопрос целостности базы данных связан с правилами, установленными у пользователя информационной системы. Например, в компании могут быть установлены такие правила:

- клиентам не разрешается размещать заказы на сумму, превышающую их лимит кредита;
- сведения о выполненных заказах хранятся в течение шести месяцев, а затем удаляются;
- нельзя выдавать читателю новые книги, пока он не вернет взятых ранее;
- каждый раз, когда продается какой-нибудь товар, для служащего, оформившего продажу, и для отдела, в котором этот служащий работает, на стоимость проданного товара увеличивается значение определенного поля, используемого для вычисления премии; в случае возврата всего или части проданного товара, значение этого поля соответствующим образом должно уменьшиться;
- каждый раз, когда приходит новая поставка, количество поставленного товара на складе увеличивается на количество товара в поставке.

Такие правила называются бизнес-правилами. В первом стандарте SQL считалось, что эти правила выходят за рамки ответственности СУБД и за их реализацию отвечает прикладная программа, осуществляющая доступ к базе данных. Впервые в 1986 году в СУБД Sybase было введено понятие триггер, что позволило включить реализацию бизнес-правил в базу данных

С любым событием, вызывающим изменение содержимого таблицы, можно связать сопутствующее действие (триггер), которое СУБД должна выполнять при каждом возникновении события. Триггер – это группа операторов языка SQL, которые автоматически выполняются при вставке, модификации или удалении записи.

В СУБД можно создавать триггеры, работающие при следующих шести условиях:

- до вставки записи (BEFORE INSERT);
- после вставки записи (AFTER INSERT);
- до удаления записи (BEFORE DELETE);
- после удаления записи (AFTER DELETE);
- до модификации записи (BEFORE UPDATE);
- после модификации записи (AFTER UPDATE).

Триггер может срабатывать при возникновении одного из нескольких событий. Триггеры могут вызывать выполнение хранимых процедур *РРТ, выполнять различные проверки и генерировать исключения. Обычно триггеры используются для задания сложных правил контроля целостности базы данных, которые невозможно реализовать с помощью ограничений

2.Алгоритм выполнения работы

1. Подключиться к базе данных.

2. Открыть окно с таблицей, для которой будет создаваться триггер.
3. Перейти на вкладку "Триггеры" (Рис. 1).
4. Нажать в этом окне правую кнопку мыши на одном из событий, для которого будет создаваться триггер.
5. Откроется контекстное меню, в котором надо выбрать команду "Новый триггер".
6. В результате откроется окно создания триггера (рис. 2 или 3), в котором достаточно лишь ввести тело триггера, а затем нажать кнопку [**Компилировать триггер**] (Ctrl+F9).
7. Переключение между двумя режимами просмотра и редактирования триггера, показанными на рис. 2 и 3 производится кнопкой [**Включить/Выключить "ленивый" режим**].
8. Создание с помощью программы "IBExpert" в диалоговом режиме генераторов и исключений происходит следующим образом:
9. Открывается окно "Генераторы" либо "Исключения". Для этого можно, на-пример, выбрать команду главного меню **База данных > Новый генератор** или **База данных > Новое исключение**.
10. В окне "Генераторы" или "Исключения" вводятся для генератора – имя и начальное значение, а для исключения – имя и текст сообщения. Можно ввести данные сразу о нескольких генераторах и исключениях.
11. После ввода данных необходимо нажать кнопку [**Компиляция**] (F9), которая приведет к автоматической генерации необходимых операторов и их выполнению.

Контрольные вопросы

1. Для чего нужны генераторы
2. Функции триггеров

Методические указания к выполнению
практического занятия № 11

Тема: Сортировка и фильтрация данных

Цель: Изучить используемые в MS Access способы сортировки и фильтрации данных.

1.Теоретическое обоснование

В Microsoft Access предусмотрено четыре способа отбора записей с помощью фильтров: фильтр по выделенному фрагменту, обычный фильтр, поле **Фильтр для** (Filter For) и расширенный фильтр.

Фильтр по выделенному фрагменту, обычный фильтр и поле **Фильтр для** (Filter For) являются очень простыми способами отбора записей, причем самым простым является фильтр по выделенному фрагменту — он позволяет найти все записи, содержащие определенное значение в выбранном поле. Обычный фильтр используется для отбора записей по значениям нескольких полей. Поле **Фильтр для** (Filter For) используется, если фокус ввода находится в поле таблицы и нужно ввести конкретное искомое значение или выражение, результат которого будет применяться в качестве условия отбора. Для создания сложных фильтров следует использовать окно расширенного фильтра.

Набор записей, которые были отобраны в процессе фильтрации, называется *результирующим набором.*

Чтобы использовать фильтр по выделенному фрагменту, необходимо:

1. В поле объекта в режиме Таблицы найти значение, которое должны содержать записи, включаемые в результирующий набор при применении фильтра.
2. Выделить это значение и нажать кнопку **Фильтр по выделенному** (Filter by Selection) на панели инструментов **Режим таблицы** (Table Datasheet).

Замечание

Фильтры сохраняются автоматически при сохранении таблицы или формы. Таким образом, при повторном открытии таблицы или формы можно снова применить сохраненный фильтр.

Фильтр позволяет отбирать записи, не содержащие выбранного значения. Для этого необходимо выбрать значение, нажать правую кнопку мыши и выполнить команду **Исключить выделенное** (Filter Excluding Selection).

2.Алгоритм выполнения работы

1. Открыть таблицу в режиме Таблицы (например, откройте таблицу "Заказы" (Orders)).
2. Нажать кнопку **Изменить фильтр** (Filter by Form) на панели инструментов **Режим таблицы** (Table Datasheet). Появится форма **фильтр** (Filter by form) — специальное окно для изменения фильтра (рис. 2.59). Форма содержит линейку полей таблицы. В любое из этих полей можно ввести или выбрать из списка значение, которое и будет являться условием отбора. Если условия ввести в несколько полей, они будут объединяться с помощью логического оператора И. Для того чтобы объединить условия по ИЛИ, нужно раскрыть другую вкладку формы, щелкнув по ярлычку Или в нижней части формы.
3. Выберите, например, значение **Ernst Handel** из списка в поле **Клиент** (Customer) и дату 1.01.98 в поле **Дата размещения.** Добавьте значок > перед датой. При этих условиях Access будет отбирать все заказы для Ernst Handel, размещенные после 1 января 1998 года.
4. Щелкните мышью по ярлычку Или и раскройте вторую вкладку.
5. Выберите из списка в поле **Клиент** (Customer) значение. **Alfreds Futterkiste,** а в поле **Дата размещения** снова введите значение > 1.01.98. Теперь будут отбираться заказы двух клиентов: Ernst Handel и Alfreds Futterkiste, размещенные после 1 января 1998 года.

Нажмите кнопку **Применение фильтра** (Apply Filter) на панели инструментов
Условия отбора записей, которые вводятся в поля формы **фильтр** (Filter by Form), можно сохранить в базе данных в виде запроса. Для этого необходимо при открытой форме **фильтр** (Filter by Form) нажать кнопку **Сохранить как запрос** (Save As Query) на
панели инструментов. Затем в диалоговом окне **Сохранение в виде запроса** (Save As Query) ввести имя запроса и нажать кнопку **ОК.** Когда потребуется повторить установку такого фильтра, нужно опять же при открытой форме **фильтр** (Filter by Form) нажать к Отобранные с помощью фильтра данные можно копировать, экспортировать и рассылать. Копирование данных из выборки осуществляется так же, как копирование данных в таблице.
Экспорт данных из выборки позволяет сохранить их в отдельном файле. Чтобы экспортировать данные, необходимо:

1. Выбрать в меню **Файл** (File) команду **Экспорт** (Export).
2. В окне **Экспорт объекта: Таблица <имя таблицы> в** (Export Table <имя таблицы> to) выбрать папку для экспорта файла.
3. В поле **Имя файла** (File Name) ввести имя файла, в который осуществляется экспорт.
4. В поле **Тип файла** (Save as Type) выбрать тип файла, в который осуществляется экспорт.
5. Нажать кнопку **Сохранить все** (Export All).

В отличие от обычного экспорта данных, рассылка позволяет сразу отправить результаты выборки по электронной почте разным адресатам для дальнейшего ознакомления с материалами, их анализа и т. п. Чтобы отослать результаты выборки, необходимо:

1. Выбрать в меню **Файл** (File) команду **Отправить** (Send To).
2. Из раскрывающегося меню выбрать пункт **Почтовый клиент** (Mail Recipient (as Attachment)).
3. В окне **Послать** (Send) выбрать тип файла.
4. Нажать кнопку ОК и далее выполнить стандартную процедуру для отправки почтового сообщения в своем почтовом клиенте.

нопку **Загрузить из запроса** (Load from Query) на панели инструментов.

Контрольные вопросы

1. Типы фильтров и особенности их применения
2. Каковы особенности использования расширенных фильтров

Методические указания к выполнению
практического занятия № 12

Тема: **Группировка данных**

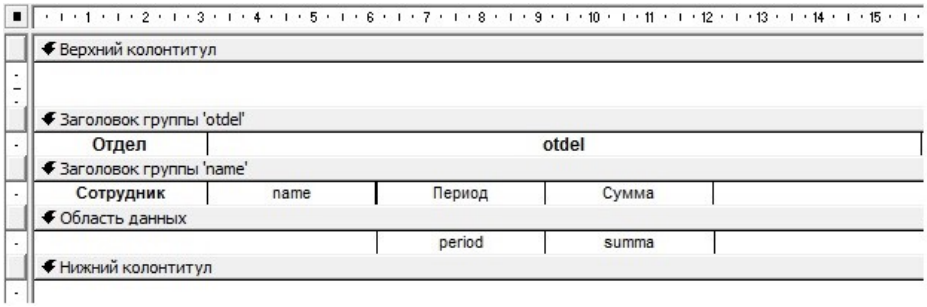
Цель: Изучить используемые в MS Access способы группировки данных данных.

1.Теоретическое обоснование

Часто для наглядности получаемых отчетов бывает нужно прибегнуть к группировкам. Например, необходимо получить данные либо по отделу в целом, либо по конкретному сотруднику.

2.Алгоритм выполнения работы

Для этого открываем в конструкторе отчета, окошко «Сортировка и группировка» выбираем там поле или поля группировки и в свойствах группы указываем показывать заголовок. Здесь мы указали, что первая группировка будет по полю otdel а вторая по полю name. И в конструкторе отчетов, после того как Вы укажите «Да» в поле показывать заголовок, у Вас сразу появятся эти самые заголовки, и Вам останется всего лишь переконструировать свой отчет, проще говоря, переставить поля таким образом, чтобы это устроил начальство, для примера я сделал следующим образом:



В итоге, когда Вы сохраните и запустите отчет, он будет уже выглядеть вот так:

Отдел	**IT**		
Сотрудник	Иванов	Период	Сумма
		Май 2014	1000
		Июнь 2014	2000
Отдел	**Финансовый**		
Сотрудник	Петров	Период	Сумма
		Май 2014	5000
		Июнь 2014	6000

Теперь более наглядно видно, какой отдел, что за сотрудник. И самое главное мы не меняли источник данных, другими словами, если дан доступ править только клиентское приложение, а доступа к серверу нет, можно легко обойтись и без этого доступа.

Контрольные вопросы

1. Что такое группировка и для чего она нужна
2. Необходим ли доступ к серверу для получения отчета с группировкой?

Методические указания к выполнению
практического занятия № 13

Тема: **Управление транзакциями**

Цель: Изучить механизм использования транзакций.

1.Теоретическое обоснование

Под управлением транзакциями понимается способность управлять различными операциями над данными, которые выполняются внутри реляционной СУБД. Прежде всего, имеется в виду выполнение операторов INSERT, UPDATE и DELETE. Например, после создания таблицы (выполнения оператора CREATE TABLE) не нужно фиксировать результат: создание таблицы фиксируется в базе данных автоматически. Точно так же с помощью отмены транзакции не удастся восстановить только что удаленную оператором DROP TABLE таблицу.

После успешного выполнения команд, заключенных в тело одной транзакции, немедленного изменения данных не происходит. Для окончательного завершения транзакции существуют так называемые команды управления транзакциями, с помощью которых можно либо сохранить в базе данных все изменения, произошедшие в ходе ее выполнения, либо полностью их отменить.

Существуют три команды, которые используются для управления транзакциями:

- COMMIT – для сохранения изменений ;
- ROLLBACK – для отмены изменений ;
- SAVEPOINT – для установки особых точек возврата.

После завершения транзакции вся информация о произведенных изменениях хранится либо в специально выделенной оперативной памяти, либо во временной области отката в самой базе данных до тех пор, пока не будет выполнена одна из команд управления транзакциями. Затем все изменения или фиксируются в базе данных, или отбрасываются, а временная область отката освобождается.

Команда COMMIT предназначена для сохранения в базе данных всех изменений, произошедших в ходе выполнения транзакции. Она сохраняет результаты всех операций, которые имели место после выполнения последней команды COMMIT или ROLLBACK .

Команда ROLLBACK предназначена для отмены транзакций, еще не сохраненных в базе данных. Она отменяет только те транзакции, которые были выполнены с момента выдачи последней команды COMMIT или ROLLBACK .

Команда SAVEPOINT (точка сохранения) предназначена для установки в транзакции особых точек, куда в дальнейшем может быть произведен откат (при этом отката всей транзакции не происходит).Команда имеет следующий вид:

```
SAVEPOINT имя_точки_сохранения
```

Она служит исключительно для создания точек сохранения среди операторов, предназначенных для изменения данных. Имя точки сохранения в связанной с ней группе транзакций должно быть уникальным.

Для отмены действия группы транзакций, ограниченных точками сохранения, используется команда ROLLBACK со следующим синтаксисом:

```
ROLLBACK TO имя_точки_сохранения
```

Поскольку с помощью команды SAVEPOINT крупное число транзакций может быть разбито на меньшие и поэтому более управляемые группы, ее применение является одним из способов управления транзакциями.

2.Алгоритм выполнения работы

Явные транзакции требуют, чтобы пользователь указал начало и конец транзакции, используя следующие команды:

- начало транзакции: в журнале транзакций фиксируются первоначальные значения изменяемых данных и момент начала транзакции ;
- BEGIN TRAN[SACTION]
- [имя_транзакции |
- @имя_переменной_транзакции
  [WITH MARK ['описание_транзакции']]]

- конец транзакции: если в теле транзакции не было ошибок, то эта команда предписывает серверу зафиксировать все изменения, сделанные в транзакции, после чего в журнале транзакций помечается, что изменения зафиксированы и транзакция завершена;

```
COMMIT [TRAN[SACTION]
        [имя_транзакции |
@имя_переменной_транзакции]]
```

- создание внутри транзакции точки сохранения: СУБД сохраняет состояние БД в текущей точке и присваивает сохраненному состоянию имя точки сохранения;

```
SAVE TRAN[SACTION]
  {имя_точки_сохранеия |
 @имя_переменной_точки_сохранения}
```

- прерывание транзакции ; когда сервер встречает эту команду, происходит откат транзакции, восстанавливается первоначальное состояние системы и в журнале транзакций отмечается, что транзакция была отменена. Приведенная ниже команда отменяет все изменения, сделанные в БД после оператора BEGIN TRANSACTION или отменяет изменения, сделанные в БД после точки сохранения, возвращая транзакцию к месту, где был выполнен оператор SAVE TRANSACTION.

```
ROLLBACK [TRAN[SACTION]
  [имя_транзакции |
  @имя_переменной_транзакции
  | имя_точки_сохранения
|@имя_переменной_точки_сохранения]]
```

Функция @@TRANCOUNT возвращает количество активных транзакций.

Функция @@NESTLEVEL возвращает уровень вложенности транзакций.

```
BEGIN TRAN
SAVE TRANSACTION  point1
```

16.1. Использование точек сохранения

В точке point1 сохраняется первоначальное состояние таблицы Товар

```
DELETE FROM Товар WHERE КодТовара=2
    SAVE TRANSACTION  point2
```

В точке point2 сохраняется состояние таблицы Товар без товаров с кодом 2.

```
DELETE FROM Товар WHERE КодТовара=3
    SAVE TRANSACTION  point3
```

В точке point3 сохраняется состояние таблицы Товар без товаров с кодом 2 и с кодом 3.

```
DELETE FROM Товар WHERE КодТовара<>1
    ROLLBACK TRANSACTION point3
```

Происходит возврат в состояние таблицы без товаров с кодами 2 и 3, отменяется последнее удаление.

```
SELECT * FROM Товар
```

Оператор SELECT покажет таблицу Товар без товаров с кодами 2 и 3.

```
ROLLBACK TRANSACTION point1
```

Происходит возврат в первоначальное состояние таблицы.

```
SELECT * FROM Товар
COMMIT
```

Первоначальное состояние сохраняется