Образовательное частное учреждение высшего образования
«ГУМАНИТАРНО-СОЦИАЛЬНЫЙ ИНСТИТУТ»

УТВЕРЖДЕНО
заседанием Ученого совета
протокол № 7 от 27.06.2023 г.
приказ ректора об утв. ОП ВО
№ 01-03/70 П от 28.06.2023 г.
Ректор _____ Л.Ф. Логинова

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

Б1.В.01 «БАЗЫ ДАННЫХ»

Код и направление подготовки:

38.03.05 «Бизнес-информатика»

Направленность (профиль):
«**Информационная бизнес-аналитика**»

Красково - 2023

Рабочая программа учебной дисциплины разработана на основе Федерального государственного образовательного стандарта высшего образования (далее – ФГОС ВО) по программе подготовки 38.03.05 «Бизнес-информатика».

Организация – разработчик: Образовательное частное учреждение высшего образования «Гуманитарно-социальный институт».

Разработчики:

д.п.н., проф.		Фураев А.Н.
ученая степень, звание	*подпись*	*ФИО*
ученая степень, звание	*подпись*	*ФИО*

Рабочая программа учебной дисциплины утверждена на заседании кафедры «Общеобразовательных дисциплин» от 22.06.2023 г. протокол №10

Заведующий кафедрой
Д.ф.н., профессор _____ Кузнецова Т.Ф.
подпись

Наименование дисциплины **– Базы данных**

1. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

Цель и **задачи** освоения дисциплины заключаются в систематизации, закреплении и расширении полученных теоретических знаний и практических умений по созданию и сопровождению информационных систем, автоматизирующих задачи организационного управления и бизнес-процессы.

Дисциплина «Базы данных» в рамках воспитательной работы направлена на формирование у обучающихся: психологической готовности к профессиональной деятельности по избранной профессии; воспитание у обучающихся уважения к труду, людям труда, трудовым достижениям и подвигам; потребности трудиться, добросовестного, ответственного и творческого отношения к разным видам трудовой деятельности; развитие навыков высокой работоспособности и самоорганизации, гибкости, умение действовать самостоятельно, активно и ответственно, мобилизуя необходимые ресурсы, правильно оценивая смысл и последствия своих действий; коммуникативной культуры и развитие органов студенческого самоуправления; исследовательского и критического мышления у обучающихся; повышение мотивации к научно-исследовательской деятельности, интереса к науке в целом; развитие творческой культуры и эрудиции; формирование навыков творческого применения на практике достижений научного прогресса; развитие навыков решения прикладных задач с использованием научных методов, продвижение собственных научных идей.

Планируемые результаты обучения

Процесс освоения дисциплины направлен на формирование следующих компетенций:

ПК-2 Способен выполнять работы по созданию (модификации) и сопровождению информационных систем, автоматизирующих задачи организационного управления и бизнес-процессы.

Подготовка по дисциплине реализуется на основе профессионального стандарта ПС 06.015 «Специалист по информационным системам».

Матрица связи дисциплины Б1.В.01 «Базы данных» и компетенций, формируемых на основе изучения дисциплины, с временными этапами освоения ее содержания

Код и наименование компетенции выпускника	**Код и наименование индикатора компетенции выпускника**	**Код индикатора компетенции выпускника**	**Код и наименование дескрипторов (планируемых результатов обучения выпускников)**
ПК-2. Способен выполнять работы по созданию (модификации) и сопровождению информационных систем, автоматизирующих задачи организационного управления и бизнес-процессы	**ПК-2.2.** Разрабатывает прототипы информационной системы на базе типовой ИС и осуществляет кодирование на языках программирования	**ПК-2.2.**	**ПК-2.2.1** ***Знать:*** основы современных систем управления базами данных; теорию баз данных; языки современных бизнес-приложений; принципы работы с запросами SQL; **ПК-2.2.2** ***Уметь:*** принимать решение о пригодности архитектуры; разрабатывать код ИС и баз данных ИС; выполнять верификацию кода ИС и баз данных ИС относительно дизайна ИС и структуры баз данных ИС; устранять обнаруженные несоответствия; выполнять работу с запросами SQL; **ПК-2.2.3** ***Владеть:*** навыками, приемами и технологиями кодирования на языках программирования; навыками создания баз данных и обработки данных в БД посредством SQL-запросов;

2. Место учебной дисциплины в структуре образовательной программы

Дисциплина части, формируемая участниками образовательных отношений.

Для изучения данной дисциплины необходимы общие знания, умения и навыки, формируемые предшествующими дисциплинами.

В структурной форме межпредметные связи изучаемой дисциплины указаны в соответствии с учебным планом образовательной программы по очной форме обучения.

Связь дисциплины «Базы данных» с последующими дисциплинами и сроками их изучения

Код дисциплины	*Дисциплины, с последующими сроками изучения «Базы данных»*	*Семестр*
Б1.О.31	Информационная безопасность	5
Б1.В.04	Основы программирования в ИС	5
Б1.В.06	Автоматизация бизнес-процессов	5,6
Б1.В.09	Конфигурирование и моделирование в системе "1С: Предприятие"	6
Б1.В.ДВ.02.01	Системы электронного документооборота	6
Б1.В.ДВ.02.02	Системы управления корпоративным контентом	6
Б1.В.ДВ.03.01	ИСУ предприятием ("1С: Предприятие")	8
Б1.В.ДВ.03.02	Программирование в 1С:	8
Б2.О.03(П)	Производственная практика: технологическая (проектно-технологическая) практика	6
Б2.О.04(Пд)	Производственная практика: преддипломная практика	8
Б2.В.01(П)	Производственная практика: практика по получению профессиональных умений и опыта профессиональной деятельности	7

3. Объем дисциплины в зачетных единицах с указанием количества академических часов, выделенных на контактную работу обучающихся с преподавателем (по видам учебных занятий) и на самостоятельную работу

Виды учебной работы	**Форма обучения**	
	Очная	**Очно-заочная**
Порядковый номер семестра	4	5
Общая трудоемкость дисциплины всего (в з.е):	6	6
Контактная работа с преподавателем всего (в акад. часах), в том числе:	93	69
Занятия лекционного типа (лекции)	34	34
Лабораторные работы	18	10
Занятия семинарского типа (практические занятия, семинары в том числе в форме практической подготовки)	34	18
Текущая аттестация	1	1
Курсовой проект (работа)	2	2
Консультации (предэкзаменационные)	2	2
Промежуточная аттестация	2	2
Самостоятельная работа всего (в акад. часах), в том числе:	123	147
Форма промежуточной аттестации:		
зачет/ дифференцированный зачет		
экзамен	Экзамен, курсовой проект (работа)	Экзамен, курсовой проект (работа)
Общая трудоемкость дисциплины (в акад. часах)	216	216

4. Содержание дисциплины, структурированное по темам (разделам)

4.1. Тематическое планирование

Тема 1. Архитектура базы данных. Основные понятия и определения.
Тема 2. Процесс прохождения пользовательского запроса.
Тема 3. Пользователи банков данных.
Тема 4. Классификация моделей данных. Теоретико-графовые модели данных. Иерархическая модель данных. Сетевая модель данных.
Тема 5. Реляционная модель данных. Основы реляционной алгебры. Операции над множествами. Специальные операции. Инфологическая модель предметной области. Проектирование реляционных БД на основе принципов нормализации. Системный анализ предметной области.
Тема 6. ER-диаграмма. Нормальные формы ER-диаграмм. Даталогические модели. Получение реляционной схемы из ER-диаграммы. Физические модели. Проектирование реляционной базы данных. Универсальное отношение. Пример проектирования реляционной БД.
Тема 7. Введение в SQL. Основные понятия и компоненты. Инструкции и имена. Типы данных. Встроенные функции. Значения NULL. Ограничения целостности.
Тема 8. Управление таблицами и данными. Команда создания таблицы — CREATE TABLE. Изменение структуры таблицы — команда ALTER TABLE. Извлечение данных — команда SELECT. Раздел SELECT. Раздел FROM. Раздел WHERE. Раздел ORDER BY. Раздел GROUP BY. Раздел COMPUTE. Раздел UNION. Раздел INTO. Использование команды SELECT...INTO

4.2. Содержание лабораторных работ по дисциплине

№	Содержание лабораторных работ	Виды практических занятий	Текущий контроль
1.	**Тема 1 Знакомство с СУБД MS Access.** Базы данных как средство хранения и обработки информации *(Приложение 1)*	устный опрос по теме практического занятия, выполнение практического задания	Практическое задание Индивидуальное задание
2.	**Тема 2. Построение ER-диаграммы.** Разработка ER-модели предметной области. Приобретение навыков моделирования предметной области, построения диаграмм «сущность-связь» *(Приложение 1)*	устный опрос по теме практического занятия, выполнение практического задания	Практическое задание Индивидуальное задание
3.	**Тема 3. Построение базы данных в СУБД Access. Нормализация отношений.** Построение базы данных по образцу. Работа с отношениями. *(Приложение 1)*	устный опрос по теме практического занятия, выполнение	Практическое задание Индивидуальное задание

		практического задания	
4.	**Тема 4. Создание запросов.** Запросы. *(Приложение 1)*	устный опрос по теме практического занятия, выполнение практического задания	Практическое задание Индивидуальное задание
5.	**Тема 5. Введение ограничений целостности базы данных в СУБД Access.** Методы обеспечения целостности. *(Приложение 1)*	устный опрос по теме практического занятия, выполнение практического задания	Практическое задание Индивидуальное задание
6.	**Тема 6. Разработка информационной системы для работы с базой данных.** Базы данных сети Интернет. Возможности PHP, Apache, MySQL. *(Приложение 1)*	устный опрос по теме практического занятия, выполнение практического задания	Практическое задание Индивидуальное задание
7.	**Тема 7. «Создание SQL-запросов».** Создание SQL-запросов. *(Приложение 1)*	устный опрос по теме практического занятия, выполнение практического задания	Практическое задание Индивидуальное задание
8.	**Тема 8. «Создание концептуальной модели данных в среде автоматизированного проектирования».** *(Приложение 1)*	устный опрос по теме практического занятия, выполнение практического задания	Практическое задание Индивидуальное задание

4.3. Самостоятельная работа студента

№	Наименование темы дисциплины	Формы подготовки
1.	**Тема 1**. Архитектура базы данных. Основные понятия и определения.	Самостоятельное изучение теоретических разделов курса.
2.	**Тема 2. Процесс прохождения пользовательского запроса.**	Самостоятельное изучение теоретических разделов курса.
3.	**Тема 3. Пользователи банков данных.**	Самостоятельное изучение теоретических разделов курса.
4.	**Тема 4. Классификация моделей данных.** Теоретико-графовые модели	Самостоятельное изучение теоретических разделов курса.

	данных. Иерархическая модель данных. Сетевая модель данных.	
5.	**Тема 5. Реляционная модель данных.** Основы реляционной алгебры. Операции над множествами. Специальные операции. Инфологическая модель предметной области. Проектирование реляционных БД на основе принципов нормализации. Системный анализ предметной области.	Самостоятельное изучение теоретических разделов курса.
6.	**Тема 6. ER-диаграмма.** Нормальные формы ER-диаграмм. Даталогические модели. Получение реляционной схемы из ER-диаграммы. Физические модели. Проектирование реляционной базы данных. Универсальное отношение. Пример проектирования реляционной БД.	Самостоятельное изучение теоретических разделов курса.
7.	**Тема 7. Введение в SQL.** Основные понятия и компоненты. Инструкции и имена. Типы данных. Встроенные функции. Значения NULL. Ограничения целостности.	Самостоятельное изучение теоретических разделов курса.
8.	**Тема 8. Управление таблицами и данными.** Команда создания таблицы — CREATE TABLE. Изменение структуры таблицы — команда ALTER TABLE. Извлечение данных — команда SELECT. Раздел SELECT. Раздел FROM. Раздел WHERE. Раздел ORDER BY. Раздел GROUP BY. Раздел COMPUTE. Раздел UNION. Раздел INTO. Использование команды SELECT...INTO	Самостоятельное изучение теоретических разделов курса.

А) Реферат– это продукт самостоятельной работы студента, представляющий собой краткое изложение в письменном виде полученных результатов теоретического анализа определенной научной (учебно-исследовательской) темы, где автор раскрывает суть исследуемой проблемы, приводит различные точки зрения, а также собственные взгляды на нее.

Примерные темы рефератов:

1. История развития, назначение и роль баз данных.
2. Файловые системы и базы данных.
3. Структуры данных и базы данных.
4. Способы хранения информации в базах данных.
5. Способы повышения эффективности обработки данных за счет их организации.
6. Общая характеристика, назначение, возможности, состав и архитектура СУБД.
7. Классификация СУБД.
8. Информационное, лингвистическое, математическое, аппаратное, организационное, правовое обеспечения СУБД.
9. Типология баз данных. Документальные базы данных. Фактографические базы данных.

10. Типология баз данных. Гипертекстовые и мультимедийные базы данных.
11. Типология баз данных. Объектно-ориентированные базы данных.
12. Типология баз данных. Распределенные базы данных. Коммерческие базы данных.
13. Недостатки реляционных СУБД.
14. Объектные расширения реляционных СУБД.
15. Средства автоматизации проектирования баз данных.
16. Централизация логики приложения на сервере базы данных.
17. Информационные хранилища. OLAP-технология.
18. XML-серверы.
19. Принципы построения БД.
20. Проблема создания и сжатия больших информационных массивов, информационных хранилищ и складов данных.
21. Фрактальные методы в архивации.
22. Управление складами данных.
23. Средства поддержания целостности базы данных
24. Серверы баз данных.
25. Многоплатформеные СУБД. СУБД Oracle.
26. Многоплатформеные СУБД. Informix.
27. Многоплатформеные СУБД. Sybase.
28. Многоплатформеные СУБД. DB2.
29. Многоплатформеные СУБД. MySQL.
30. СУБД, ориентированные на конкретные платформы. СУБД DBManager в OS/2.
31. СУБД, ориентированные на конкретные платформы. СУБД SQL/400 в AS/400.
32. СУБД, ориентированные на конкретные платформы. СУБД Access в Microsoft Windows.
33. СУБД семейства XBase, Dbase.
34. Базы данных реального времени.
35. Жизненный цикл базы данных.
36. Циклическая база данных.
37. Сжатие без потерь в реляционных СУБД.
38. Защита информации в СУБД.
39. Нормальные формы: НФБК. 3 примера.
40. Нормальные формы: 4НФ. 3 примера.
41. Нормальные формы: 5НФ. Описание. 3 примера.
42. Хранение деревьев в реляционных базах данных.
43. Способы переноса данных с одного типа БД в другую. На примере переноса данных из MySQL в Access.
44. Способы переноса данных с одного типа БД в другую. На примере переноса данных из Access в MySQL.
45. Экспорт/импорт между базами данных различных производителей.
46. Реальные и фантастические разработки БД.
47. Физическое хранение реляционных таблиц.
48. Сериализация транзакций в БД.
49. Анализ качества баз данных.
50. Пути формирования баз данных для директ-маркетинга.
51. Архитектура и функционирование адресных баз данных.

52. Сверхбольшие базы данных.
53. Эксплуатация баз данных. Состав, порядок планирования и проведения регламентных работ.
54. Эксплуатация баз данных. Сервисные средства СУБД.
55. Эксплуатация баз данных. Задачи администратора базы данных.
56. Эксплуатация баз данных. Организация труда обслуживающего персонала.

Б) Практическая работа - это средство, позволяющее оценить умение обучающегося излагать суть поставленной задачи, самостоятельно применять стандартные методы решения поставленной задачи с использованием имеющихся средств и лабораторной базы, проводить анализ полученного результата работы.

Примерные практические задания:

Задание 1. Создайте папку с вашей фамилией. Скопируйте в свою папку базу данных «Студент». Постройте предложенные запросы к базе данных.
Запросы:
1. Определить и вывести номер группы, в которой учится 25 человек.
2. Определить всех студентов, поступивших в 2020 году.
3. Выбрать все фамилии, которые начинаются на «К» или «Н».
4. Подсчитать количество студентов, обучающихся в колледже.
5. Добавить в таблицу СТУДЕНТЫ новую запись.
6. Изменить фамилию в таблице «Преподаватели», преподавателя Петров на Сидоров.

Задание 2. Создайте папку с вашей фамилией. Скопируйте в свою папку базу данных «Студент». Постройте предложенные запросы к базе данных.
Запросы:
1. Вывести информацию о преподавателях, значения кода преподавателя, которых находятся между 3 и 7. Расположить в порядке убывания.
2. Определить всех студентов поступивших в 2020 году.
7. Выбрать все предметы, которые начинаются на «С» или «М».
3. Определить средний балл по успеваемости у студента с номером 32.
4. Добавить в таблицу Преподаватель новую запись.
5. Изменить в таблице УСПЕВАЕМОСТЬ оценки студента Иванов с 4 на 5.

Задание 3. Создайте папку с вашей фамилией. Скопируйте в свою папку базу данных «Студент». Постройте предложенные запросы к базе данных.
Запросы:
1. Выбрать всех преподавателей, год рождения которых больше 1968.
2. Выбрать все предметы, которые начинаются на «И» или «Ф».
3. Определить оценки студентов группы М-63 по предмету «Математика».
4. Определить количество записей в таблице Студент.
5. Добавить в таблицу предмет новую запись.
6. Изменить фамилию студента Елисеев на Черных.

Задание 4. Создайте папку с вашей фамилией. Скопируйте в свою папку базу данных «Студент». Постройте предложенные запросы к базе данных.
Запросы:
1. Вывести информацию о преподавателях 1956 года рождения.
2. Определить всех преподавателей, которые относятся к кафедре, название которой начинается на букву «Л». Расположить в порядке убывания.
3. Вывести информацию о студентах, фамилия которых начинаются на «Ф» и «Х».

4. Подсчитать средний балл, полученный студентами по предмету «Русский язык и культура речи».
5. Добавить в таблицу Группа новую запись.
6. Изменить название предмета с «СММ» на «Системы массового мониторинга»

Задание 5. Создайте папку с вашей фамилией. Скопируйте в свою папку базу данных «Студент». Постройте предложенные запросы к базе данных.
Запросы:
1. Определить номер группы, в которой учится больше 20 человек.
2. Определить всех студентов с фамилиями на букву С.
3. Определить оценки студента Мелехов.
4. Определить группу, в которой обучается максимальное количество студентов.
5. Добавить в таблицу Кафедра новую запись.
6. Изменить фамилию преподавателя Попова на Романову.

Задание 6. Создайте папку с вашей фамилией. Скопируйте в свою папку базу данных «Студент». Постройте предложенные запросы к базе данных.
Запросы:
1. Вывести информацию о студентах, значения номера, которых находятся между 44 и 69 Расположить в порядке убывания.
2. Определить всех студентов не 1999 года рождения, расположить в алфавитном порядке.
3. Определить всех студентов с фамилиями на букву М.
4. Определить средний балл по успеваемости у студента с номером 28.
5. Добавить в таблицу Преподаватель новую запись.
6. Изменить в таблице УСПЕВАЕМОСТЬ оценки студента Валуйкина с 4 на 5.

Задание 7. Создайте папку с вашей фамилией. Скопируйте в свою папку базу данных «Студент». Постройте предложенные запросы к базе данных.
Запросы:
1. Выбрать всех преподавателей, год рождения которых не 1968.
2. Определить всех студентов с фамилиями на букву З.
3. Определить количество записей в таблице Преподаватель.
4. Определить студентов, которые не сдали экзамен по предмету «Операционные системы».
5. Добавить в таблицу предмет новую запись.
6. Изменить фамилию студента Фурцев на Кудрявцева.

Задание 8. Создайте папку с вашей фамилией. Скопируйте в свою папку базу данных «Студент». Постройте предложенные запросы к базе данных.
Запросы:
1. Выбрать всех студентов, год рождения которых не 2001.
2. Определить всех студентов с фамилиями на букву И.
3. Определить количество записей в таблице Балл
4. Определить студентов, которые сдали экзамен по предмету «Операционные системы».
5. Добавить в таблицу Студент новую запись.
6. Изменить название предмета «Математика» на «Экономическая теория».

В) Собеседование – специальная беседа преподавателя со студентом на темы, связанные с изучаемой дисциплиной, рассчитанная на выяснение объема знаний студента по определенному разделу, теме, проблеме и т.п.

Примерные вопросы для собеседования:
1. Архитектура базы данных.
2. Процесс прохождения пользовательского запроса
3. Пользователи банков данных
4. Классификация моделей данных
5. Иерархическая модель данных
6. Сетевая модель данных
7. Основы реляционной алгебры. Операции над множествами.
8. Специальные операции.
9. Проектирование реляционных БД на основе принципов нормализации
10. Системный анализ предметной области
11. Инфологическая модель предметной области.
12. ER-диаграмма. Нормальные формы ER-диаграмм
13. Даталогические модели. Физические модели
14. Получение реляционной схемы из ER-диаграммы
15. Проектирование реляционной базы данных
16. Универсальное отношение
17. Основные понятия и компоненты SQL. Инструкции и имена
18. Типы данных
19. Встроенные функции. Значения NULL
20. Ограничения целостности
21. Первичный ключ таблицы
22. Внешний ключ таблицы
23. Определение уникального столбца
24. Определение проверочных ограничений
25. Определение значения по умолчанию
26. Команда CREATE TABLE
27. Команда ALTER TABLE
28. Раздел SELECT
29. Раздел FROM
30. Раздел WHERE
31. Раздел ORDER BY
32. Раздел GROUP BY
33. Раздел COMPUTE
34. Раздел UNION
35. Раздел INTO. Использование команды SELECT...INTO
36. Команда INSERT
37. Команда UPDATE
38. Команда DELETE
39. Реляционная модель данных
40. Схема прохождения запроса к БД

Г) Тест – это система стандартизированных простых и комплексных заданий, позволяющая автоматизировать процедуру измерения уровня знаний, умений и навыков обучающегося.

Примерные тестовые задания:
Тест 1.
1. Базы данных -это:
- сложная программа, направленная учет входящей информации
+ наборы данных, находящиеся под контролем систем управления
- бесконечный объем данных, постоянно управляющийся с помощью СУБД

2. Основное отличие реляционной БД:
+ данные организовываются в виде отношений
- строго древовидная структура
- представлена в виде графов

3. Расширением файла БД является:
- .f2
+ .mdb, .db
- .mcs

4. Слово Null в БД используется для обозначения:
+ неопределенных значений
- пустых значений
- нуля

5. Что такое кортеж?
- совокупность атрибутов
+ множество пар атрибутов и их значений
- схема отношений данных

6. Мощность отношений - это:
- количество веток в графовой системе
- порядок подчинения данных в древовидной структуре БД
+ количество кортежей в отношении

7. Главное условие сравнимых отношений:
+ одинаковая схема отношений
- точное количество сравнимых признаков
- наличие количественности признаков

8. Операция проекции направлена на:
- накладывание данных одной БД на данные другой БД
+ выборку данных согласно заданным атрибутам
- сравнение БД на основе схожести

9. В отличие от пользовательского типа данных базовые типы данных:
+ присутствуют в БД изначально
- должны быть в любой БД
- имеют более простую структуру

10. Если а - это цена, б - масса, то атрибут с, обозначающий стоимость будет:
- базовым атрибутом
+ виртуальным атрибутом
- сложным атрибутом

11. Подсхема исходной схемы, состоящая из одного или нескольких атрибутов, для которых декларируется условие уникальности значений в кортежах отношений называется?
- глобальная схема отношений
+ ключ
- отчет

12. Индекс для подсхемы, состоящей из нескольких атрибутов называется:

+ составной
- неуникальный
- сложный

13. В MS Access нельзя осуществить запрос на:
- обновление данных
+ создание данных
- добавление данных

14. MS Access при закрытии программы:
- предлагает сохранить БД
+ автоматически сохраняет при вводе данных
- автоматически сохраняет при закрытии программы

15. Для эффективной работы БД должно выполняться условие:
+ непротиворечивости данных
- достоверности данных
- объективности данных

16. Поле «Счетчик» отличается тем, что:
- обязательно должны вводиться целые числа
- в поле хранится только значение, а сами данные в другом поле
+ в нем происходит автоматическое наращивание

17. Какая функция позволяет выбрать несколько атрибутов сразу из нескольких таблиц и получить новую таблицу с результатом?
- форма
+ запрос
- отчет

18. Для чего предназначены формы в MS Access?
+ для ввода данных в удобном порядке
- для вывода данных в удобном формате
- для представления конечной информации в удобном виде

19. Какой символ заменяет все при запросе в БД?
+ символ *
- символ "
- символ &

20. Что позволяет автоматизировать ввод данных в таблицу?
- шаблон
- значение по умолчанию
+ список подстановки

21. Запросы создаются с помощью:
+ мастера запросов
- службы запросов
- клиента запросов

22. Основные понятия иерархической БД:
- таблица, столбец, строка

+ уровень, узел, связь
- отношение, атрибут, кортеж

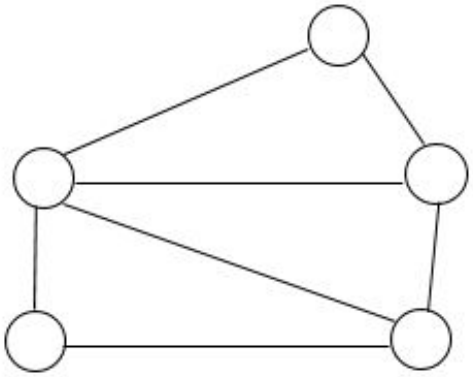
23. В чем особенность фактографической БД?
+ содержит краткие сведения об описываемых объектах, представленные в строго определенном формате
- содержит информацию разного типа
- содержит информацию определенного типа

24. Пример фактографической БД:
- законодательный акт
- приказ по учреждению
+ сведения о кадровом составе учреждения

25. Информационная система - это?
+ совокупность БД и СУБД
- комплекс аппаратно-программных средств, предназначенных для работы с инфомацией
- совокупность данных

26. Данные - это:
- представление информации в формализованном виде для работы с ними
- информация в определенном контексте
+ факты, которые не подверглись обработке

27. Какую модель данных можно изобразить графом, представленным на рисунке?



Тест 2
1. В таблицу базы данных СКЛАД, содержащую 5 столбцов информации о товаре (наименование, поставщик, количество, дата окончания срока хранения, цена), внесена информация о 25 видах товара. Количество записей в таблице равно …
Варианты ответа: а) 25 б) 5 в) 125 г) 30

2. В СУБД MS Access не существует запрос на _____ данных.
Варианты ответа: а) создание б) обновление в) удаление г) добавление

3. Реляционная база данных задана тремя таблицами. Поля Код спортсмена, Код дистанции, Дата соревнования, Время, Телефон соответственно должны иметь типы …
Варианты ответа: а) числовой (целое), текстовый, дата/время, числовой (с плавающей точкой), текстовый б) числовой (целое), текстовый, дата/время, числовой (с плавающей точкой), числовой (с плавающей точкой) в) числовой (целое), текстовый, дата, время, текстовый г) числовой (целое), текстовый, дата/время, дата/время, текстовый

4. Реляционная база данных задана тремя таблицами. Связи между таблицами могут быть установлены следующим образом: …
Варианты ответа: а) таблицы 1 и 2 связаны через поля Код дистанции, таблицы 1 и 3 связаны через поля Код спортсмена б) таблицы 1 и 2 связаны через поля Время и Рекорд, таблицы 1 и 3 связаны через поля Код спортсмена в) таблицы 1 и 2 связаны через поля Код дистанции, таблицы 1 и 3 связаны через поля Код спортсмена и Фамилия г) таблицы 1 и 2 связаны через поля Код дистанции, таблицы 1 и 3 связаны через поля Код спортсмена, таблицы 2 и 3 связаны через поля Код спортсмена и Код дистанции

5. Для первичного ключа ложно утверждение, что …
Варианты ответа: а) первичный ключ может принимать нулевое значение б) в таблице может быть назначен только один первичный ключ в) первичный ключ может быть простым и составным г) первичный ключ однозначно определяет каждую запись в таблице

6. Дан фрагмент базы данных «Тестирование»: для подсчета общего количества баллов каждого студента необходимо создать запрос …
Варианты ответа: а) с вычисляемым полем б) с параметром в) с критерием поиска г) на обновление

7. При закрытии таблицы СУБД MS Access не предлагает выполнить сохранение внесенных данных, потому что данные сохраняются …
Варианты ответа: а) автоматически сразу же после ввода в таблицу б) только после закрытия всей базы данных в) автоматически при закрытии таблицы базы данных г) после ввода пользователем специальной команды Сохранение данных
8. Представлена база данных «Тестирование». Условиям поиска удовлетворяет(-ют) _____ записей.
Варианты ответа: а) 5 б) 4 в) 2 г) 6

9. Для эффективной работы с базой данных система управления базами данных (СУБД) должна обеспечивать _____ данных.
Варианты ответа: а) непротиворечивость б) достоверность в) объективность г) кодирование

10. Дан фрагмент базы данных «Страны мира». После проведения сортировки сведения о Великобритании переместятся на одну строку вверх. Это возможно, если сортировка будет проведена в порядке …
Варианты ответа: а) убывания по полю Население б) возрастания по полю Плотность в) возрастания по полю Перепись г) убывания по полю Площадь

11. Построенная модель не должна содержать избыточную информацию.
Варианты ответа: а) наименование, количество, цена, дата окончания срока хранения б) наименование, количество, дата окончания срока хранения, общая сумма в) наименование, количество, цена, дата окончания срока хранения, текущая дата г) наименование, количество, цена, текущая дата, дата окончания срока хранения, общая сумма

12. Выбрать необходимые данные из одной или нескольких взаимосвязанных таблиц в MS Access, отобрать нужные поля, произвести вычисления и получить результат в виде новой таблицы можно с помощью …
Варианты ответа: а) запроса б) схемы данных в) главной кнопочной формы г) составной формы

13. Требуется восстановить номер телефона абонента, о котором известно, что его фамилия либо Михайлов, либо Михайловский, проживает он на Невском проспекте и номер его

телефона оканчивается на цифру 7. Соответствующий запрос должен иметь вид … Варианты ответа: а) (Фамилия = “Михайло*”)И (Адрес = “Невский проспект”)И (Телефон = ###-##-#7) б) (Фамилия = “Михайлов”)И (Адрес = “Невский проспект”)И (Телефон = ###-##-#7) в) (Фамилия = “Мих*”)И (Адрес = “Невский проспект”)И (Телефон = ###-##-#7) г) (Фамилия = “Михайло*”)И (Адрес = “Нев*”)И (Телефон = ###-##-#7)

14. Графическое отображение логической структуры базы данных в MS Access, задающее ее структуру и связи, называется …
Варианты ответа: а) схемой б) графом в) образом г) алгоритмом

15. Основными объектами СУБД MS Access являются …
Варианты ответа: а) таблица, форма, отчет, запрос б) конструктор, мастер, шаблон, схема данных в) таблица, поле, запись, ключ г) схема данных, ключ, шаблон, отчет

16. База данных, содержащая сведения о студентах, участвующих в научно-исследовательских работах (НИРС), имеет _____ структуру.
Варианты ответа: а) сетевую б) иерархическую в) древовидную г) списочную

17. Дан фрагмент базы данных «Телефонный справочник». Требуется восстановить номер телефона абонента, о котором известно, что его фамилия либо Михайлов, либо Михайловский, проживает он на Невском проспекте и номер его телефона оканчивается на цифру 7. Соответствующий запрос должен иметь вид …
Варианты ответа: а) (Фамилия = “Михайло*”)И (Адрес = “Невский проспект”)И (Телефон = ###-##-#7) б) (Фамилия = “Михайлов”)И (Адрес = “Невский проспект”)И (Телефон = ###-##-#7) в) (Фамилия = “Мих*”)И (Адрес = “Невский проспект”)И (Телефон = ###-##-#7) г) (Фамилия = “Михайло*”)И (Адрес = “Нев*”)И (Телефон = ###-##-#7)

18. Дан фрагмент базы данных «Страны мира». После проведения сортировки сведения о Великобритании переместятся на одну строку вверх. Это возможно, если сортировка будет проведена в порядке …
Варианты ответа: а) убывания по полю Население б) возрастания по полю Плотность в) возрастания по полю Перепись г) убывания по полю Площадь

19. Дан фрагмент базы данных «Склад»: после проведения сортировки сведения о товаре «Сканер планшетный» переместились на одну строку вниз. Это возможно, если сортировка проводилась по …
Варианты ответа: а) возрастанию поля «Цена, руб.» б) убыванию поля «Цена, руб.» в) возрастанию поля «Наименование» г) убыванию поля «Количество, шт.»

20. Автоматизировать операцию ввода в связанных таблицах позволяет …
Варианты ответа: а) список подстановки б) шаблон в) условие на допустимое значение г) значение по умолчанию

21. Дан фрагмент базы данных «Сотрудники». Чтобы повысить всем сотрудникам зарплату на 20%, необходимо создать запрос …
Варианты ответа: а) на обновление б) с вычисляемым полем в) с параметром г) с групповыми операциями

22. Для таблицы реляционной базы данных ложно утверждение, что …
Варианты ответа: а) каждая запись в таблице содержит однородные по типу данные б) все столбцы таблицы содержат однородные по типу данные в) в таблице нет двух одинаковых записей г) каждый столбец таблицы имеет уникальное имя

23. Средство визуализации информации, позволяющее осуществить выдачу данных на устройство вывода или передачу по каналам связи, – это …
Варианты ответа: а) отчет б) форма в) шаблон г) заставка

24. Основными понятиями иерархической структуры являются …
Варианты ответа: а) уровень, узел, связь б) отношение, атрибут, кортеж в) таблица, столбец, строка г) таблица, поле, запись

25. Для таблицы реляционной базы данных ложно утверждение, что …
Варианты ответа: а) каждая запись в таблице содержит однородные по типу данные б) все столбцы таблицы содержат однородные по типу данные в) в таблице нет двух одинаковых записей г) каждый столбец таблицы имеет уникальное имя

4.4. Распределение часов по темам и видам учебных занятий

Номер раздела, темы дисциплины	***Компетенции***	***Контактная работа***		***Лекции***		***Практические занятия Семинары***		***Лабораторные работы***		***Самост. работа студентов***	
Формы обучения		***ОФО***	***ОЗФО***	***ОФО***	***ОЗФО***	***ОФО***	***ОЗФО***	***ОФО***	***ОЗФО***	***ОФО***	***ОЗФО***
Тема 1	ПК-2	6	12	2	6	2	4	2	2	11	11
Тема 2	ПК-2	10		4		4		2		10	12
Тема 3	ПК-2	10	14	4	8	4	4	2	2	10	12
Тема 4	ПК-2	10		4		4		2		10	14
Тема 5	ПК-2	10	8	4	4	4	2	2	2	12	16
Тема 6	ПК-2	10	16	4	10	4	4	2	2	12	16
Тема 7	ПК-2	16		6		6		4		12	16
Тема 8	ПК-2	14	12	6	6	6	4	2	2	12	16
Текущая аттестация	ПК-2	1									
Консультации (предэкзаменационные)	ПК-2	2									
Подготовка, консультации, прием защиты курсовых проектов (работ)	ПК-2	2								34	34
Промежуточная аттестация	ПК-2	2									
Всего:		***93***	***69***	***34***	***34***	***34***	***18***	***18***	***10***	***123***	***147***

4.5. Методические указания для обучающихся по освоению дисциплины

Для правильной организации самостоятельной работы необходимо учитывать порядок изучения разделов курса, находящихся в строгой логической последовательности. Поэтому хорошее усвоение одной части дисциплины является предпосылкой для успешного перехода к следующей. Для лучшего запоминания материала целесообразно использовать индивидуальные особенности и разные виды памяти: зрительную, слуховую, ассоциативную. Успешному запоминанию способствует приведение ярких свидетельств и наглядных примеров. Учебный материал должен постоянно повторяться и закрепляться.

Подготовка к практическому (семинарскому) занятию начинается с тщательного ознакомления с условиями предстоящей работы, т. е. с обращения к вопросам семинарских занятий. Определившись с проблемой, следует обратиться к рекомендуемой литературе. При подготовке к практическому (семинарскому) занятию обязательно требуется изучение дополнительной литературы по теме занятия. Без использования нескольких источников информации невозможно проведение дискуссии на занятиях, обоснование собственной позиции, построение аргументации. Если обсуждаемый аспект носит дискуссионный характер, следует изучить существующие точки зрения и выбрать тот подход, который вам кажется наиболее верным. При этом следует учитывать необходимость обязательной аргументации собственной позиции. Во время практических занятий рекомендуется активно участвовать в обсуждении рассматриваемой темы, выступать с подготовленными заранее докладами и презентациями, принимать участие в выполнении практических заданий.

С целью обеспечения успешного обучения студент должен готовиться к лекции, поскольку она является важной формой организации учебного процесса: знакомит с новым учебным материалом; разъясняет учебные элементы, трудные для понимания; систематизирует учебный материал; ориентирует в учебном процессе.

Подготовка к лекции заключается в следующем:

- внимательно прочитайте материал предыдущей лекции;
- узнайте тему предстоящей лекции (по тематическому плану);
- ознакомьтесь с учебным материалом по учебным пособиям;
- постарайтесь уяснить место изучаемой темы в своей профессиональной подготовке;
- запишите возможные вопросы, которые вы зададите преподавателю на лекции.

Во время лекции рекомендуется составлять конспект, фиксирующий основные положения лекции и ключевые определения по пройденной теме.

К экзамену необходимо готовится целенаправленно, регулярно, систематически и с первых дней обучения по дисциплине. Попытки освоить дисциплину в период зачётно-экзаменационной сессией, как правило, показывают не слишком хороший результат. В самом начале учебного курса студенту следует познакомиться со следующей учебно-методической документацией:

- программой дисциплины;
- перечнем знаний и умений, которыми студент должен овладеть;
- тематическими планами лекций, семинарских занятий;
- контрольными мероприятиями;
- учебными пособиями по дисциплине;
- перечнем экзаменационных вопросов.

После этого у студента должно сформироваться четкое представление об объеме и характере знаний и умений, которыми надо будет овладеть по дисциплине. Систематическое выполнение учебной работы на лекциях, семинарских занятиях и в процессе самостоятельной работы позволит успешно освоить дисциплину и создать хорошую базу для сдачи экзамена.

Рекомендуемая тематика занятий максимально полно реализуется в контактной работе со студентами очной формы обучения. В случае реализации образовательной программы в заочной / очно-заочной форме трудоемкость дисциплины сохраняется, однако объем учебного материала в значительной части осваивается студентами в форме самостоятельной работы. При этом требования к ожидаемым образовательным результатам студентов по данной дисциплине не зависят от формы реализации образовательной программы.

В случае организации учебной работы с использованием дистанционных образовательных технологий занятия проводятся в электронной информационно-образовательной среде института.

5. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ

5.1 Образовательные технологии

В освоении учебной дисциплины «Базы данных» используются следующие **традиционные образовательные технологии:**

- чтение информационных лекций с использованием доски и видеоматериалов;
- практические/семинарские занятия;
- лабораторные занятия;
- контрольные опросы;
- консультации;
- самостоятельная работа студентов;
- тестирование по основным темам дисциплины;
- защита курсового проекта (работы);
- экзаменационная аттестация.

5.2.Использование информационных технологий:

- технологии, основанные на использовании ЭИОС института (методические материалы по дисциплине, размещенные на сайте ГСИ);
- Интернет-технологии;
- компьютерные обучающие и контролирующие программы;
- информационные технологии, позволяющие увеличить эффективность преподавания (за счет усиления иллюстративности):
  - ***лекция-визуализация –*** *иллюстративная форма проведения информационных и проблемных лекций;*
  - ***семинар-презентация*** *– использование студентами на семинарах специализированных программных средств.*

5.3. Активные и интерактивные методы и формы обучения

Из перечня видов: («мозговой штурм», анализ проблемных ситуаций, анализ конкретных ситуаций, инциденты, имитация коллективной профессиональной деятельности, творческая

работа, связанная с самопознанием и освоением дисциплины, деловая игра, круглый стол, диспут, дискуссия, мини-конференция и др.) используются следующие:

- «*мозговой штурм»;*
- *диспут* (способ ведения спора, проводимого с целью установления научной истины со ссылками на устоявшиеся письменные авторитетные источники и тщательный анализ аргументов каждой из сторон);
- *дискуссия (*как метод, активизирующий процесс обучения, изучения сложной темы, теоретической проблемы*) применяется на семинарах-дискуссиях, где обсуждаются спорные вопросы с выявлением мнений в студенческой группе;*
- *беседа.*

6. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине

Контроль качества освоения дисциплины включает в себя текущий контроль успеваемости и промежуточную аттестацию обучающихся. Промежуточная аттестация обучающихся по дисциплине проводится в форме экзамена и курсового проекта (работы).

Конкретный перечень типовых контрольных заданий и иных материалов для оценки результатов освоения дисциплины, а также описание показателей и критериев оценивания компетенций приведен в фонде оценочных средств по дисциплине.

6.1. Формы текущего контроля

- *индивидуальное собеседование;*
- *выполнение тестовых заданий;*
- *выполнение рефератов;*
- *выполнение курсового проекта;*
- *мониторинг результатов практических занятий;*
- *мониторинг результатов лабораторных занятий.*

6.2. Форма промежуточного контроля по дисциплине – экзамен.

Вопросы к *экзамену:*

(теоретическая часть):

1. Архитектура базы данных.
2. Процесс прохождения пользовательского запроса
3. Пользователи банков данных
4. Классификация моделей данных
5. Иерархическая модель данных
6. Сетевая модель данных
7. Основы реляционной алгебры. Операции над множествами.
8. Специальные операции.
9. Проектирование реляционных БД на основе принципов нормализации
10. Системный анализ предметной области
11. Инфологическая модель предметной области.
12. ER-диаграмма. Нормальные формы ER-диаграмм
13. Даталогические модели. Физические модели

14. Получение реляционной схемы из ER-диаграммы
15. Проектирование реляционной базы данных
16. Универсальное отношение
17. Основные понятия и компоненты SQL. Инструкции и имена
18. Типы данных
19. Встроенные функции. Значения NULL
20. Ограничения целостности
21. Первичный ключ таблицы
22. Внешний ключ таблицы
23. Определение уникального столбца
24. Определение проверочных ограничений
25. Определение значения по умолчанию
26. Команда CREATE TABLE
27. Команда ALTER TABLE
28. Раздел SELECT
29. Раздел FROM
30. Раздел WHERE
31. Раздел ORDER BY
32. Раздел GROUP BY
33. Раздел COMPUTE
34. Раздел UNION
35. Раздел INTO. Использование команды SELECT...INTO
36. Команда INSERT
37. Команда UPDATE
38. Команда DELETE
39. Реляционная модель данных
40. Схема прохождения запроса к БД

(Практические задания)

6.3 Требования к выполнению курсового проекта (работы) находятся в Методических рекомендациях по выполнению курсовых проектов (работ).

Примерные темы курсового проекта:

1. Сетевые базы данных и СУБД
2. Иерархические базы данных и СУБД
3. Реляционные базы данных и СУБД
4. Объектные базы данных и СУБД
5. Объектно-ориентированные базы данных и СУБД
6. Объектно-реляционные базы данных и СУБД
7. Архитектура систем управления базами данных (СУБД)
8. Система управления базами данных MS Access
9. Система управления базами данных MS SQL Server
10. Системы управления базами данных (СУБД): настройка, обработка запросов, оптимизация
11. Системы управления базами данных (СУБД): управление параллельным доступом, транзакции, способы решения проблем
12. Распределенные и параллельные базы данных.
13. Архитектура «клиент-сервер»

14. Методология проектирования БД: моделирование предметной области
15. Методология проектирования БД: концептуальное проектирование
16. Методология проектирования БД: логическое проектирование
17. Методология проектирования БД: физическое проектирование
18. Проектирование реляционной базы данных
19. Нормализация реляционной модели данных
20. Инфологическое моделирование.
21. Модель ER «сущность-связь»
22. Принципы построения: реляционной базы данных: реляционная алгебра и реляционное исчисление
23. Язык структурированных запросов SQL: использование языка SQL в прикладном программировании
24. Язык структурированных запросов SQL: оптимизация запросов в SQL
25. Язык структурированных запросов SQL: организация защиты данных с помощью SQL
26. Безопасность и администрирование БД
27. Распределенные базы данных: технология обработки данных OLAP
28. Распределенные базы данных: технология OLTP
29. Принципы построения распределенных хранилищ данных
30. Восстановление данных в СУБД MS Access и СУБД MS SQL Server
31. Проблемы параллелизма в СУБД
32. Проблемы безопасности баз
33. Проблемы целостности баз данных.
34. Проблемы создания и сжатия больших объемов информационных массивов, информационных хранилищ и складов данных
35. Разработка и сопровождение баз данных в MS SQL Server
36. Мультимедийные базы данных
37. Графические базы данных

7. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ

7.1. Рекомендуемая литература

Основная литература

Балдин, К. В. Информационные системы в экономике : учебник / К. В. Балдин, В. Б. Уткин. — 9-е изд., стер. — Москва : Дашков и К°, 2021. — 395 с. : ил., табл. — ISBN 978-5-394-04038-2. — Текст : электронный // Университетская библиотека ONLINE : [сайт]. — URL: https://biblioclub.ru/index.php?page=book&id=684194

Советов, Б. Я. Базы данных : учебник для вузов / Б. Я. Советов, В. В. Цехановский, В. Д. Чертовской. — 3-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2023. — 420 с. — (Высшее образование). — ISBN 978-5-534-07217-4. — Текст : электронный // Образовательная платформа Юрайт : [сайт]. — URL: https://urait.ru/bcode/510752

Стружкин, Н. П. Базы данных: проектирование : учебник для вузов / Н. П. Стружкин, В. В. Годин. — Москва : Издательство Юрайт, 2023. — 477 с. — (Высшее образование). — ISBN 978-5-534-00229-4. — Текст : электронный // Образовательная платформа Юрайт : [сайт]. — URL: https://urait.ru/bcode/511019

Дополнительная литература

Гордеев, С. И. Организация баз данных : в 2 ч. Часть 1 : учебник для вузов / С. И. Гордеев, В. Н. Волошина. — 2-е изд., испр. и доп. — Москва : Издательство Юрайт, 2023. — 310 с. — (Высшее образование). — ISBN 978-5-534-04469-0. — Текст : электронный // Образовательная платформа Юрайт : [сайт]. — URL: https://urait.ru/bcode/513879

Гордеев, С. И. Организация баз данных : в 2 ч. Часть 2 : учебник для вузов / С. И. Гордеев, В. Н. Волошина. — 2-е изд., испр. и доп. — Москва : Издательство Юрайт, 2023. — 513 с. — (Высшее образование). — ISBN 978-5-534-04470-6. — Текст : электронный // Образовательная платформа Юрайт : [сайт]. — URL: https://urait.ru/bcode/515097

Илюшечкин, В. М. Основы использования и проектирования баз данных : учебник для вузов / В. М. Илюшечкин. — Москва : Издательство Юрайт, 2023. — 213 с. — (Высшее образование). — ISBN 978-5-534-03617-6. — Текст : электронный // Образовательная платформа Юрайт : [сайт]. — URL: https://urait.ru/bcode/510473

Нестеров, С. А. Базы данных : учебник и практикум для вузов / С. А. Нестеров. — Москва : Издательство Юрайт, 2023. — 230 с. — (Высшее образование). — ISBN 978-5-534-00874-6. — Текст : электронный // Образовательная платформа Юрайт : [сайт]. — URL: https://urait.ru/bcode/511650

Стружкин, Н. П. Базы данных: проектирование. Практикум : учебное пособие для вузов / Н. П. Стружкин, В. В. Годин. — Москва : Издательство Юрайт, 2023. — 291 с. — (Высшее образование). — ISBN 978-5-534-00739-8. — Текст : электронный // Образовательная платформа Юрайт : [сайт]. — URL: https://urait.ru/bcode/512160

Периодическая литература (библиотека ГСИ)

1. Проблемы управления.
2. Российский журнал менеджмента.
3. Системный администратор.
4. Программные продукты и системы (доступный архив: 2010–2022). – URL: https://www.iprbookshop.ru/25852.html.

7.2. Электронные образовательные и информационные ресурсы

1. Электронно-библиотечная система «ЮРАЙТ» - https://urait.ru/
2. Университетская библиотека онлайн – www/biblioclub.ru

7.3. Профессиональные базы данных и информационные справочные системы

Информационно-справочные системы

1. «Система КонсультантПлюс» – компьютерная справочная правовая система - http://www.consultant.ru/
2. «Гарант» – справочно-правовая система по законодательству Российской Федерации - http://www.garant.ru/
3. Единое окно доступа к образовательным ресурсам. - http://window.edu.ru/
4. Национальная информационно-аналитическая система Российский индекс научного цитирования (РИНЦ). - https://www.elibrary.ru Федеральный портал «Российское образование» - http://www.edu.ru/

Профессиональные базы данных

1. Научная электронная библиотека eLibrary.ru - Российский индекс научного цитирования (РИНЦ)
2. Открытый портал информационных ресурсов (научных статей, сборников работ и монографий по различным направлениям наук) https://elibrary.ru/project_risc.asp
3. База данных научных журналов на русском и английском языке ScienceDirect

4. Открытый доступ к метаданным научных статей по различным направлениям наук поиск рецензируемых журналов, статей, глав книг и контента открытого доступа http://www.sciencedirect.com/
5. Портал «Psychology-OnLine.Net»
6. Федеральный портал «Российское образование» http://www.edu.ru/
7. Бесплатная электронная библиотека онлайн «Единое окно доступа к образовательным ресурсам»
8. http://window.edu.ru/
9. Единая коллекция цифровых образовательных ресурсов Научно-практические и методические материалы http://school-collection.edu.ru/
10. Библиотека. Тематические подборки статей. http://www.flogiston.ru/
11. Сайт, посвященный SQL, программированию, базам данных, разработке информационных систем https://www.sql.ru/
12. На сайте проекта OpenNet размещается информация о Unix системах и открытых технологиях для администраторов, программистов и пользователей http://www.opennet.ru/
13. Библиотека программиста https://proglib.io
14. Сообщество IT-Специалистов https://habr.com/ru/
15. Сеть разработчиков Microsoft https://msdn.microsoft.com/ru-ru/
16. Сборник статей по информационной безопасности http://www.iso27000.ru/chitalnyi-zai
17. Профессиональная база данных «Всероссийская система данных о компаниях и бизнесе «За честный бизнес» // Режим доступа: https://zachestnyibiznes.ru
18. Профессиональная база данных Росстата // Режим доступа http://www.gks.ru/wps/wcm/connect/rosstat_main/rosstat/ru/statistics/databases/

Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины

1. Министерство образования и науки Российской Федерации. 100% доступ - http://минобрнауки.рф/
2. Федеральная служба по надзору в сфере образования и науки. 100% доступ - http://obrnadzor.gov.ru/
3. Федеральный портал «Российское образование». 100% доступ - http://www.edu.ru/
5. Федеральный центр информационно-образовательных ресурсов. 100% доступ - http://fcior.edu.ru/
6. Электронно-библиотечная система, содержащая полнотекстовые учебники, учебные пособия, монографии и журналы в электронном виде 5100 изданий открытого доступа. 100% доступ - http://bibliorossica.com/
7. Федеральная служба государственной статистики. 100% доступ - http://www.gks.ru

8. Программное обеспечение, используемое при осуществлении образовательного процесса по дисциплине

Операционная система Windows 10,
Microsoft office (Word, Excel, PowerPoint, Outlook, Publisher)
Microsoft Access,
Microsoft SQL Server Express Edition
SQL Server Management Studio
XAMPP
DB Designer Fork
1 С Предприятие (версия 8.3)

Антивирус Windows Defender (входит в состав операционной системы Microsoft Windows)

Программное обеспечение отечественного производства

INDIGO
Яндекс.Браузер

Свободно распространяемое программное обеспечение

Adobe Reader для Windows
Архиватор HaoZip

9. Материально-техническое обеспечение, необходимое для осуществления образовательного процесса по дисциплине

Для проведения учебных занятий используются учебные аудитории, оснащенные оборудованием и техническими средствами обучения: специализированной мебелью, отвечающей всем установленным нормам и требованиям; ПК с доступом к сети Интернет, магнитно-маркерной доской, переносным мультимедийным оборудованием, сканером, сетевым принтером.

Для самостоятельной работы обучающихся используются помещения, оснащенные компьютерной техникой: персональные компьютеры с доступом к сети Интернет и ЭИОС института, принтеры; специализированной мебелью, отвечающей всем установленным нормам и требованиям.

Для обучения инвалидов и лиц с ограниченными возможностями здоровья институтом могут быть представлены специализированные средства обучения, в том числе технические средства коллективного и индивидуального пользования.

10. Методические рекомендации по обучению лиц с ограниченными возможностями здоровья

Профессорско-педагогический состав знакомится с психолого-физиологическими особенностями обучающихся инвалидов и лиц с ограниченными возможностями здоровья, индивидуальными программами реабилитации инвалидов (при наличии). При необходимости осуществляется дополнительная поддержка преподавания тьюторами, психологами, социальными работниками, прошедшими подготовку ассистентами.

В соответствии с методическими рекомендациями Минобрнауки РФ (утв. 8 апреля 2014 г. N АК-44/05вн) в курсе предполагается использовать социально-активные и рефлексивные методы обучения, технологии социокультурной реабилитации с целью оказания помощи в установлении полноценных межличностных отношений с другими студентами, создании комфортного психологического климата в студенческой группе. Подбор и разработка учебных материалов производятся с учетом предоставления материала в различных формах: аудиальной, визуальной, с использованием специальных технических средств и информационных систем.

Освоение дисциплины лицами с ОВЗ осуществляется с использованием средств обучения общего и специального назначения (персонального и коллективного использования). Материально-техническое обеспечение предусматривает приспособление аудиторий к нуждам лиц с ОВЗ.

Форма проведения аттестации для студентов-инвалидов устанавливается с учетом индивидуальных психофизических особенностей. Для студентов с ОВЗ предусматривается доступная форма предоставления заданий оценочных средств, а именно:

- в печатной или электронной форме (для лиц с нарушениями опорно-двигательного аппарата);
- в печатной форме или электронной форме с увеличенным шрифтом и контрастностью (для лиц с нарушениями слуха, речи, зрения);
- методом чтения ассистентом задания вслух (для лиц с нарушениями зрения).

Студентам с инвалидностью увеличивается время на подготовку ответов на контрольные вопросы. Для таких студентов предусматривается доступная форма предоставления ответов на задания, а именно:

- письменно на бумаге или набором ответов на компьютере (для лиц с нарушениями слуха, речи);
- выбором ответа из возможных вариантов с использованием услуг ассистента (для лиц с нарушениями опорно-двигательного аппарата);
- устно (для лиц с нарушениями зрения, опорно-двигательного аппарата).

При необходимости для обучающихся с инвалидностью процедура оценивания результатов обучения может проводиться в несколько этапов.

Приложение 1

Лабораторные работы

Лабораторная работа № 1.

Тема: Знакомство с СУБД MS Access.

Цель работы: изучить и закрепить на практике методы и средства СУБД по корректному заполнению и модификации таблиц БД и методы контроля вводимых данных путем связывания таблиц.

Теоретическая часть

Базы данных — это совокупность сведений (о реальных объектах, процессах, событиях или явлениях), относящихся к определенной теме или задаче, организованная таким образом, чтобы обеспечить удобное представление этой совокупности, как в целом, так и любой ее части.

Почти все современные системы основаны на реляционной (relational) модели управления базами данных. Реляционная база данных представляет собой множество взаимосвязанных таблиц, каждая из которых содержит информацию об объектах определенного типа. Название «реляционная» связано с тем, что каждая запись содержит информацию, относящуюся только к одному объекту. В таких базах данные не дублируются, а связываются по определенным полям.

Можно выделить три основные функции СУБД:

-определение данных (Data definition) - вы можете определить, какая именно информация будет храниться в вашей базе данных, задать структуру данных и их тип (например, максимальное количество цифр или символов), а также указать, как эти данные связаны между собой. В некоторых случаях вы можете также задать форматы и критерии проверки данных;

-обработка данных (Data manipulation) - данные можно обрабатывать самыми различными способами. Можно объединять данные с другой связанной с ними информацией и вычислять итоговые значения;

-управление данными (Data control) - вы можете указать, кому разрешено знакомиться с данными, корректировать их или добавлять новую информацию. Можно также определить правила коллективного пользования данными.

Система управления базами данных Microsoft Access является одним из самых популярных приложений в семействе настольных СУБД.

Запуск Microsoft Access. Создание базы данных

Для того, чтобы запустить Microsoft Access необходимо:

1. Нажать кнопку Пуск на Панели задач в нижней части рабочего стола.
2. Открыть в главном меню пункт Программы.
3. Выбрать программу Microsoft Access.

Выбрать пустой шаблон действие 1 задать имя файла действие 2 и нажать кнопку создать.

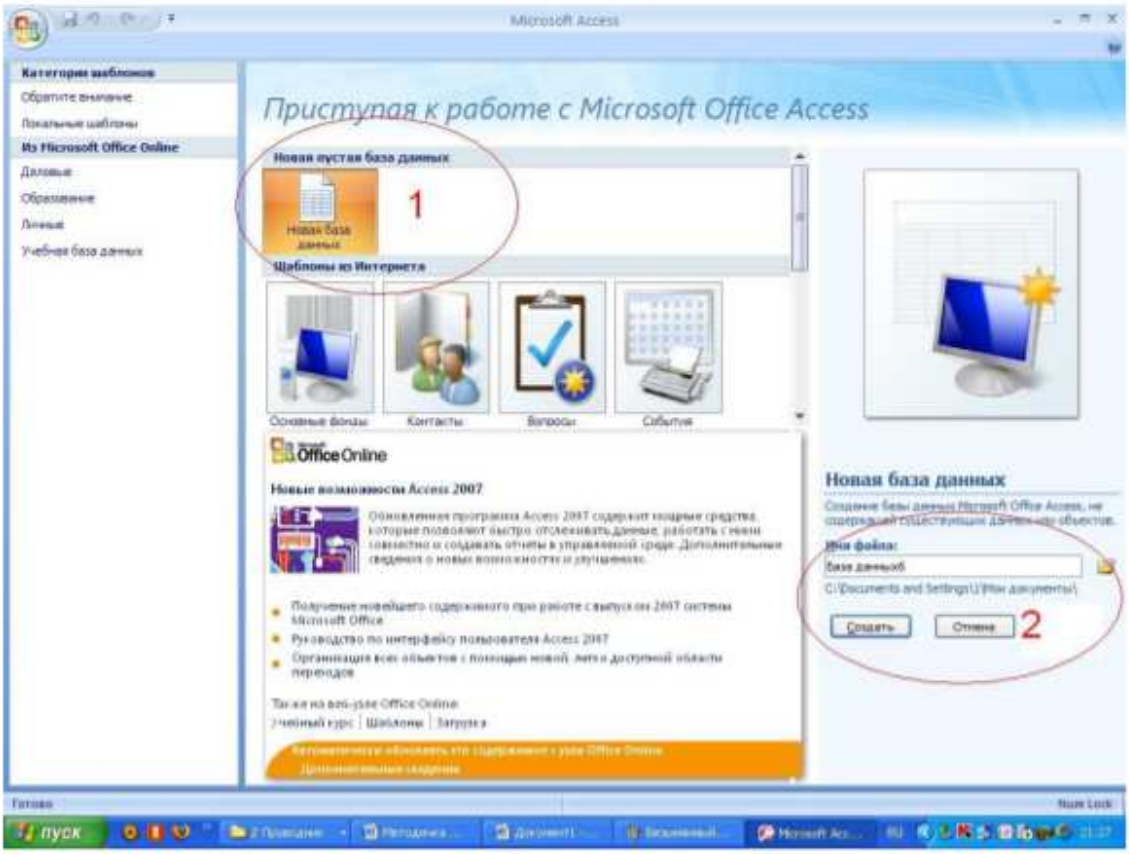


Рис.1. Создание базы данных

Под базой данных (БД) понимают хранилище структурированных данных, при этом данные должны быть непротиворечивы, минимально избыточны и целостны. Реляционные БД представляют связанную между собой совокупность таблиц-сущностей базы данных (ТБД). Связь между таблицами может находить свое отражение в структуре данных, а может только подразумеваться, то есть присутствовать на неформализованном уровне. Каждая таблица БД представляется как совокупность строк и столбцов, где строки соответствуют экземпляру объекта, конкретному событию или явлению, а столбцы - атрибутам (признакам, характеристикам, параметрам) объекта, события, явления. При практической разработке БД таблицы-сущности зовутся таблицами, строки экземпляры - записями, столбцы-атрибуты - полями ТБД. Одно из важнейших достоинств реляционных баз данных состоит в том, что можно хранить логически сгруппированные данные в разных таблицах и задавать связи между ними, объединяя их в единую базу. Такая организация данных позволяет уменьшить избыточность хранимых данных, упрощает их ввод и организацию запросов и отчетов. Понятие первичного ключа В каждой таблице БД может существовать первичный ключ. Под первичным ключом понимают поле или набор полей, однозначно (уникально) идентифицирующих запись. Первичный ключ должен быть минимально достаточным: в нем не должно быть полей, удаление которых из первичного ключа не отразится на его уникальности.

Таблица 1. Данные отношения «преподаватель»

Таб. №	ФИО	Уч. степень	Уч. звание	Код кафедры
101	Иванов И.И	Д-р тех. наук	Профессор	01
102	Петров С.В.	Канд. техн. наук	Доцент	01
103	Конин Е.А	Канд. техн. наук	Доцент	01
104	Глухова А.А.	Канд. техн. наук	Доцент	01

В качестве первичного ключа в таблице «Преподаватель» может выступать только «Таб. №», значения других полей могут повторяться внутри данной таблицы. Правила нормализации баз данных, и чисто практические соображения должны побудить разработчика всегда определять первичный ключ для таблицы базы данных. Реляционные отношения (связи) между таблицами базы данных. Между двумя или более таблицами базы данных могут существовать отношения подчиненности. Отношения подчиненности определяют, что для каждой записи главной таблицы {master, называемой еще родительской} может существовать одна или несколько записей в подчиненной таблице {detail, называемой еще дочерней}. Существует три разновидности связей между таблицами базы данных:

- «один-ко-многим»,
- «один-к-одному»,
- «многие-ко-многим». Отношение «один-ко-многим» имеет место, когда одной записи родительской таблицы может соответствовать несколько записей в дочерней таблице. Связь «один-ко-многим» является самой распространенной для реляционных баз данных. В широко распространенной нотации структуры баз данных IDEF1X отношение «один-ко многим» изображается путем соединения таблиц линией, которая на стороне дочерней таблицы оканчивается кружком или иным символом. Поля, входящие в первичный ключ для данной ТБД, всегда расположены вверху и отчеркнуты от прочих полей линией.

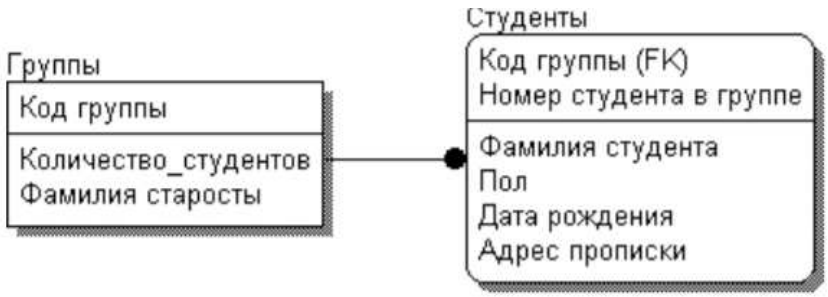


Рис.2. Отношение один-ко многим

Отношение «один-к-одному» имеет место, когда одной записи в родительской таблице соответствует одна запись в дочерней таблице.

№ сотрудника	ФИО	Должность	Отдел
1	Иванов П.П.	лаборант	2
2	Сидорова А.М.	бухгалтер	1
3	Петров А.Н.	инженер	2

№ сотрудника	Год рождения	Число детей
1	1967	1
2	1955	1
3	1943	2

Рис.3. Отношение один-к одному

Данное отношение используют, если не хотят, чтобы таблица БД «не распухала» от второстепенной информации. Отношение «многие-ко-многим» имеет место, когда: а) записи в родительской таблице может соответствовать больше одной записи в дочерней таблице; б) записи в дочерней таблице может соответствовать больше одной записи в родительской таблице. Например, каждой студент изучает несколько дисциплин. Каждая дисциплина изучается несколькими студентами.

Реляционные СУБД (в частности Access) не поддерживают связи «многие-ко-многим» на уровне индексов и ссылочной целостности. Считается, что всякую связь «многие-ко-многим» можно заменить на одну или более связей «один-ко-многим».

Например,

Рис.5. Трансформации связи многие-ко многим

Ссылочная целостность и каскадные воздействия Рассмотрим наиболее часто встречающуюся в базах данных связь «один-ко-многим». Как можно заметить, дочерняя и родительская таблицы связаны между собой по общему полю «Шифр группы». Назовем это поле полем связи.

Таблица «Группа»

Шифр группы	Кол-во студ-в	Прох. балл
101	30	4,50
102	32	4,50
103	29	4,80

Таблица «Студент»

Шифр группы	№ студ.	ФИО	Год рожд.	Прох. балл
101	01	Аристов Р.П.	1979	4,25
101	02	Борисова С.А.	1979	4,25
102	01	Макова Н.В.	1978	4,75

Рис.6. Отношение один-ко многим

Возможны два вида изменений, которые приведут к утере связей между записями в родительской и дочерней таблицах:

- изменение значения поля связи в записи родительской таблицы без изменения значений полей связи в соответствующих записях дочерней таблицы;
- изменение значения поля связи в одной из записей дочерней таблицы без соответствующего изменения значения полей связи в родительской и дочерней таблицах.

Таблица «Группа»

Шифр группы	Кол-во студ-в	Прох. балл
А-101	30	4,50
102	32	4,50
103	29	4,80

Таблица «Студент»

Шифр группы	№ студ.	ФИО	Год рожд.	Прох. балл
101	01	Аристов Р.П.	1979	4,25
101	02	Борисова С.А.	1979	4,25
102	01	Макова Н.В.	1978	4,75

Таблица «Группа»

Шифр группы	Кол-во студ-в	Прох. балл
101	30	4,50
102	32	4,50
103	29	4,80

Таблица «Студент»

Шифр группы	№ студ.	ФИО	Год рожд.	Прох. балл
А-101	01	Аристов Р.П.	1979	4,25
101	02	Борисова С.А.	1979	4,25
102	01	Макова Н.В.	1978	4,75

Рис.7. Примеры нарушения целостности базы данных

И в первом, и втором случаях наблюдается нарушение целостности базы данных, поскольку информация в ней становится недостоверной. Следовательно, нужно блокировать действия, которые нарушают целостность связей между таблицами, которую называют ссылочной целостностью. Чтобы предотвратить потерю ссылочной целостности, используется механизм каскадных изменений. Он состоит в обеспечении следующих требований:

- необходимо запретить изменение поля связи в записи дочерней таблицы без синхронного изменения полей связи в родительской таблице;
- при изменении поля связи в записи родительской таблице, следует синхронно изменить значения полей связи в соответствующих записях дочерней таблицы;
- при удалении записи в родительской таблице, следует удалить соответствующие записи в дочерней таблице. Необходимость разрешения или запрещения каскадных изменений обычно реализуется в СУБД при определении связей между таблицами. Собственно, таким образом, и происходит создание ссылочной целостности. Понятие внешнего ключа Для обеспечения ссылочной целостности в дочерней таблице создается внешний ключ. Во внешний ключ входят поля связи дочерней таблицы. Для связей типа «один-ко-многим» внешний ключ по составу полей должен совпадать с первичным ключом родительской таблицы. Индексы и методы доступа Индексы представляют собой механизмы быстрого доступа к данным в таблицах БД. Сущность индексов состоит в том, что они хранят значения индексных поле (т.е. полей, по которым построен индекс) и указатель на запись в таблице. При последовательном методе доступа для выполнения запроса к таблице БД просматриваются все записи таблицы, от первой до последней. При индексно-последовательном методе доступа для выполнения запроса к таблице БД указатель в индексе устанавливается на первую строку, удовлетворяющую условию запроса (или его части), и считывается запись из таблицы по хранящемуся на нее в индексе указателю. Определение первичных и внешних ключей таблиц БД приводят к созданию индексов по полям, объявленным в составе первичных или внешних ключей. Нормализация таблиц при проектировании БД При проектировании структуры новой БД определяют сущности (объекты, явления) предметной области, которые должны найти свое отражение в базе данных. В конечном итоге анализ предметной области должен привести к созданию эскиза БД. Сначала желательно изобразить сущности и связи между ними. Как правило, каждой сущности в БД соответствует таблица. Затем - в эскизе второго порядка - для каждой таблицы БД приводится список атрибутов - полей записи. На этом этапе процесс проектирования структур БД является процессом творческим, неоднозначным, с другой стороны, узловые его моменты могут быть формализованы. Одной из таких формализаций является требование, согласно которому реляционная база данных должна быть нормализована. Процесс нормализации имеет своей целью устранение избыточности данных и заключается в приведении к третьей нормальной форме (3НФ). Первая нормальная форма (1НФ) требует, чтобы каждое поле таблицы БД:
- было неделимым;
- не содержало повторяющихся групп. Неделимость поля означает, что значение поля не должно делиться на более мелкие значения. Например, если в поле «Подразделение» содержится название факультета и название кафедры, требование неделимости не соблюдается и необходимо из данного поля выделить или название факультета, или кафедры в отдельное поле. Повторяющимися являются поля, содержащие одинаковые по смыслу значения. Например, если требуется получить статистику сдачи экзаменов по предметам, можно создать поля для хранения данных об оценке по каждому предмету. Однако в этом случае мы имеем дело с повторяющимися группами. Вторая нормальная форма (2НФ) требует, чтобы все поля таблицы зависели от первичного ключа, то есть, чтобы первичный ключ однозначно определял запись и не был избыточен. Те поля, которые зависят только от части первичного ключа, должны быть выделены в составе отдельных таблиц. Третья нормальная форма (3НФ) требует, чтобы значение любого поля таблицы, не входящего в первичный ключ, не зависело от значения другого поля, не входящего в первичный ключ.

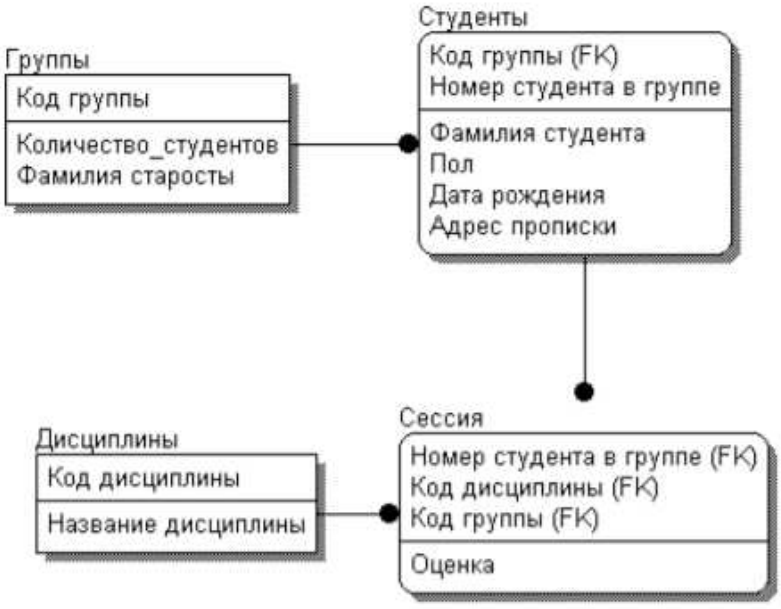


Рис.7. Пример логической модели базы данных "Сессия"

Основным средством хранения информации в СУБД Access являются плоские таблицы, состоящие из строк (записей) и именованных столбцов (полей). Каждое поле записи содержит одну характеристику объекта и имеет строго определенный тип данных (например, текстовая строка, число, дата). Оборудование и материалы (перечень используемого оборудования) – соответствующее программное обеспечение. Указания по технике безопасности: к выполнению лабораторных работ допускаются студенты, ознакомившиеся с правилами работы в лаборатории, прошедшие инструктаж безопасности. Задание. Спроектировать базу данных согласно вариантам:

Вариант 1.

Деятельность торговой фирмы. В базе данных учесть следующие признаки: дату, количество, наименование, тип, цену проданного товара, покупателя, его фирму, город, телефон.

Вариант 2.

Деятельность предприятия по сборке изделий. В базе данных учесть следующие признаки: наименование, тип, цену продажи некоторого изделия, количество дней на его сборку, количество компонент в изделии, описание, изготовитель компонент, тип и стоимость каждого компонента.

Вариант 3.

Деятельность стола заказов. В базе данных учесть следующие признаки: дату получения и исполнения заказа, скидку на заказ, количество и цену товара, вошедшего в заказ, имя клиента, его расчетный счет и величину кредита.

Вариант 4.

Оплата коммунальных услуг. В базе данных учесть следующие признаки: фамилию квартиросъемщика, его адрес, жилую площадь, число проживающий в квартире, дату и период оплаты коммунальных услуг, стоимость одного квадратного метра жилплощади, стоимость потребления холодной воды на одного проживающего.

Вариант 5.

Работа фирмы с поставщиками. В базе данных учесть следующие признаки: дату продажи некоторого товара, количество, цену, скидку при продаже и налог на продажу, а также поставщиков товара, страну и наличие лицензии на продажу.

Вариант 6.

Начисление зарплаты. В базе данных учесть следующие признаки: фамилию, адрес, телефон сотрудника, дату его рождения и дату устройства на работу, дату, вид и количество в часах выполненной работы, описание выполненной работы, тип освобождения от налога, нижнюю и верхнюю границы оплаты одного часа.

Вариант 7.

Деятельность бюро добрых услуг. В базе данных учесть следующие признаки: вид услуги, ее описание и стоимость, дату оказания этой услуги, скидку при оплате в зависимости от социального положения клиента, имя и место проживания клиента.

Вариант 8.

Оплата междугородних телефонных разговоров. В базе данных учесть следующие признаки: дату и время, продолжительность телефонного разговора, город, с которым состоялся разговор, фамилию, адрес, номер телефона клиента, тарифы городов и скидки на время разговора в течение суток.

Вариант 9.

Поваренная книга. В базе данных учесть следующие признаки: названия и типы блюд, описание компонент блюда с указанием количества в граммах, калорийности и стоимости 1 грамма, количества жиров, углеводов и белков в 1 грамме компонента.

Вариант 10.

Книжная палата. В базе данных учесть следующие признаки: дату и количество проданных книг, название, автора, издательство, тематику, цену проданной книги, сведения об авторе: фамилию, пол, дату рождения.

Вариант 11.

Музыкальная коллекция. В базе данных учесть следующие признаки: дату, количество и стоимость проданного альбома, страну, авторов слов и музыки, исполнителя, длительность каждой композиции в альбоме.

Вариант 12.

Видеотека. В базе данных учесть следующие признаки: дату продажи видеокассеты, название фильма, страну, режиссера, тематику фильма, наличие Оскаров, дату выпуска фильма, стоимость кассеты, информацию о покупателе: возраст, пол, социальное положение.

Вариант 13.

Олимпийские игры. В базе данных учесть следующие признаки: номер, символ олимпиады, город проведения, наличие в городе гор или моря. дату открытия и закрытия, число видов спорта, по которым проводятся соревнования, команды-участницы, количество спортсменов в команде, число завоеванных золотых, серебряных и бронзовых медалей каждой командой.

Вариант 14.

Учебный процесс. В базе данных учесть следующие признаки: фамилии студентов, дату рождения, курс, дату сдачи, оценку и название предмета для каждого студента, для каждого предмета указать число часов на изучение, код предмета: гуманитарный блок, математический или профессиональный и кафедру, которая ведет данный предмет.

Вариант 15.

Учебная нагрузка преподавателя. В базе данных учесть следующие признаки: фамилию, должность, звание преподавателя, кафедру, на которой он работает, название предмета, который он ведет, для предмета указать название, длительность в часах, код предмета: гуманитарный блок, математический или профессиональный, для каждой должности и звания указать стоимость часа.

Вариант 16.

Продажа билетов на самолеты. В базе данных учесть следующие признаки: дату продажи билета, номер рейса, дату вылета рейса, номер места, фамилию пассажира, его социальное положение, данные по типу самолета, обслуживающего рейс: тип самолета, стоимость билета, конечный пункт, продолжительность маршрута, экипаж, квалификация командира, его возраст и стаж полетов.

Вариант 17.

Автобусный парк. В базе данных учесть следующие признаки: дату подачи и дату исполнения заявки, продолжительность и вид поездки, число участников, сумма аванса, выплаченного за поездку, марка выделенного автобуса, его техническое состояние, количество мест, стоимость билета, налоги и скидки в зависимости от вида поездки.
Вариант 18.
Финансовое состояние вузов. В базе данных учесть следующие признаки: число госбюджетных и хоздоговорных студентов в каждой студенческой группе, число преподавателей на факультете, среднюю стоимость обучения одного студента на факультете, среднюю зарплату преподавателя по университету, название университета, город, фамилию ректора, объем госбюджетных поступлений и дотаций из местного бюджета для каждого университета.
Вариант 19.
Областное УВД. В базе данных учесть следующие признаки: дату совершения и дату раскрытия преступления, вид и тяжесть преступления, описать участников: фамилию, дату рождения, вид участия, описать примененное оружие: марку, страну изготовления, за кем числится.
Вариант 20.
Фирма по продаже подержанных автомобилей. В базе данных учесть следующие признаки: дату продажи, продавца, вид оплаты, данные о покупателе: фамилию, пол, возраст, социальное положение, информацию об автомобиле: марка, цвет, изготовитель, дата изготовления, техническое состояние, мощность двигателя.

Лабораторная работа № 2.
Тема: Построение ER-диаграммы.
Цель работы: разработка ER-модели предметной области. Приобретение навыков моделирования предметной области, построения диаграмм «сущность-связь».
Теоретическая часть
Построение базы данных предметной области «Расписание экзаменов». Проектируемая система должна выполнять следующие действия:
- Хранить информацию о студентах, предметах, экзаменах (по какому предмету, какой группой и когда сдается), группах.
- Печатать расписание для каждой группы, списки групп, результаты экзаменов. Разработаем ER-модель данной предметной области.
Примечание: исключительно ради рассмотрения различных вариантов связей и атрибутов примем за истину следующие условия:
1) дата рождения студента может быть не указана;
2) один и тот же экзамен сразу могут сдавать несколько групп. Студент, предмет, группа, экзамен - явные кандидаты на сущность.
Связи между сущностями:
- «каждый студент должен обучаться в одной группе», «группа может состоять из нескольких студентов»,
- «каждая группа может сдавать несколько экзаменов», «каждый экзамен может проводиться у одной или нескольких групп»,
- «каждый экзамен должен сдаваться по одному предмету», «по каждому предмету могут сдаваться несколько экзаменов».

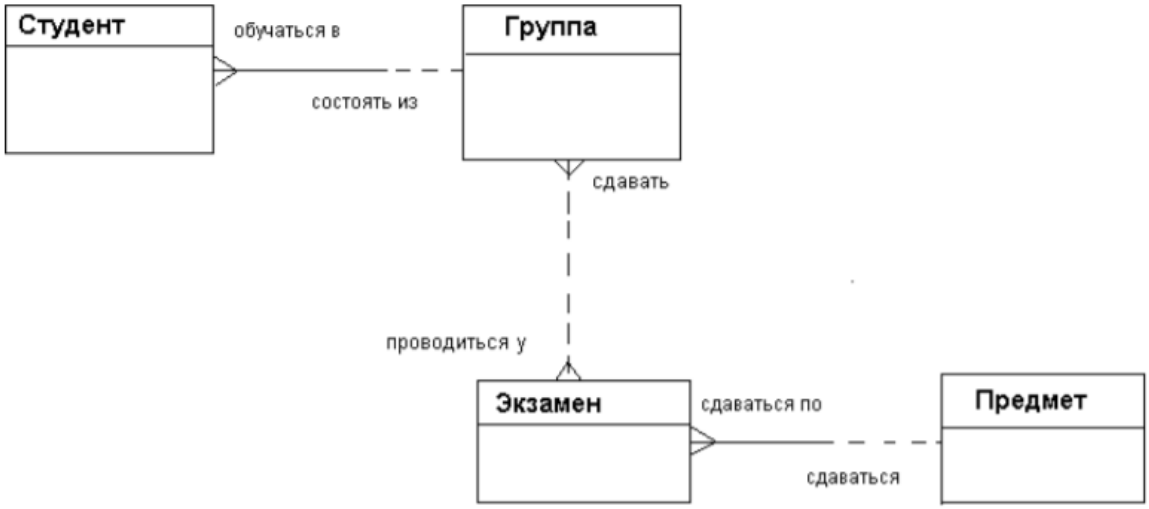


Рис.8. Пример логической модели базы данных

Атрибуты сущностей:

1. Атрибутами студента могут быть фамилия, имя, отчество, дата рождения.

2. Атрибутами группы могут быть номер группы и форма обучения (дневная или заочная).

3. Атрибутом экзамена может быть дата его сдачи.

4. Атрибутом предмета может быть его название.

Однако оценки за экзамен не могут быть атрибутом никакой из имеющихся сущностей, а, повидимому, представляют собой отдельную сущность.

Связи:

- «каждый студент может получить несколько оценок», «каждая оценка должна быть поставлена одному студенту»,
- «каждый экзамен может оценивать несколько оценок», «каждая оценка должна быть поставлена за один экзамен»

5. Атрибутом оценки может быть ее значение.

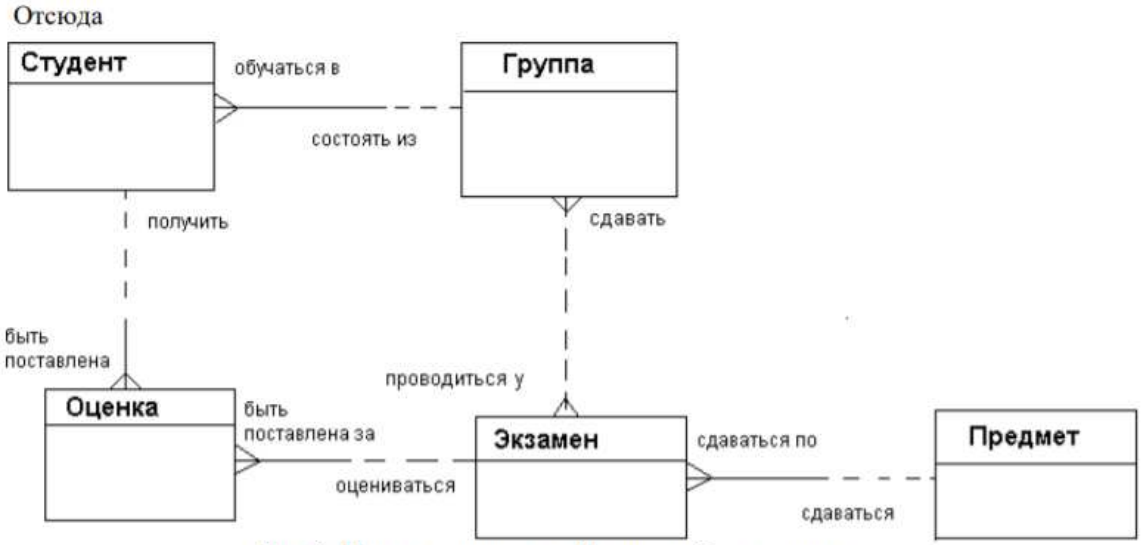


Рис.9. Пример логической модели базы данных

Сущности группа и экзамен связаны друг с другом отношением много-ко-многим. Такая связь должна быть разделена на две связи типа один ко многим. Для этого требуется дополнительная сущность. Введем сущность «Строка расписания экзаменов».

Связи:

- «каждая строка расписания экзаменов должна включать одну группу», «каждая группа может упоминаться в одной или нескольких строках расписания»;
- «каждая строка расписания экзаменов должна включать один экзамен», «каждый экзамен может упоминаться в одной или нескольких строках расписания».

Отсюда:

Рис.10. Пример логической модели базы данных

То есть мы получаем концептуальную ER-диаграмму.

Рис.11. Пример концептуальной ER-диаграммы

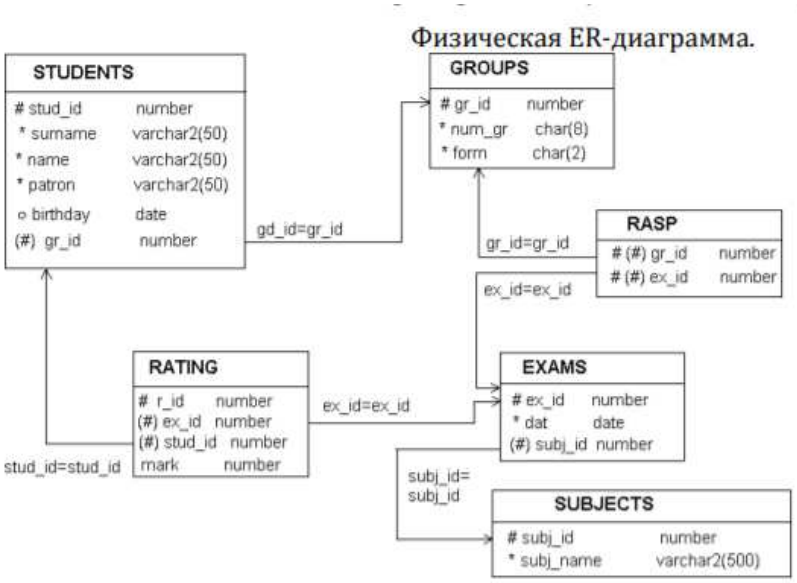


Рис.12. Пример физической модели базы данных

ER-диаграммы рекомендуется разрабатывать в среде онлайн-редактора деловой графики https://www.draw.io. Допускается также использование CASE-средства AllFusion ERwin Data Modeler, редактора деловой графики MS Visio.

Задание.

1. Институт (деканаты, кафедры, учебный отдел).

- 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.

2. Библиотека института.

- 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.

3. Отдел кадров и бухгалтерия 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.

4. 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).

5. Технологический отдел некоторого предприятия.

- Производственные процессы: продукты, объёмы их производства, необходимые материалы, количества разных видов материалов на единицу продукции, отходы производства; используемое оборудование и его тип, даты ввода оборудования в строй, сроки амортизации, производительность оборудования; человеческие ресурсы (сколько всего и сколько по производству единицы продукции — сколько необходимо и сколько реально).

-Материалы: тип (категория), марка, является ли сырьем (или производится на предприятии), потребляемые объемы (в т.ч. на единицу конечной продукции), в рамках каких технологических процессов используется, цена.

6. Отдел продаж некоторой фирмы.

- Клиенты: название компании, ФИО контактного лица, адрес выставления счёта, адрес доставки, телефон, факс.

- Заказы: тип заказа (покупка, гарантийный ремонт, негарантийный ремонт), общая стоимость, скидка, товар(ы), их изготовители, модели (марки), серийные номера, описание неисправностей, необходимые ресурсы, клиент, дата получения заказа, срок завершения, дата выставления счёта и его оплаты, метод оплаты, дата поставки, метод и стоимость доставки.

- Ресурсы: ФИО, отдел(ы) и телефон(ы) исполнителя(ей), число рабочих часов для выполнения заказа, ставка зарплаты, ответственный за выполнение заказа, необходимое оборудование и расходные материалы, их количество и стоимость, а также наличие материалов на складе.

7. Магазин (внутренний учет).

- Клиенты: юридическое или физическое лицо, ФИО, адрес, телефон, адрес выставления счёта, вид и номер карточки, факс.

- Продажи: наименования, модели (марки) и серийные номера товаров, поставка из магазина или со склада, количество и общая стоимость товаров, размер скидки, тип скидки, форма оплаты (наличными, оплата счёта, по карточке), необходимость доставки, стоимость и тип доставки, адрес доставки.

Товары: категория, модель, название производителя, адрес производителя, цена, количество в магазине и на складе.

8. Электронный магазин (информация для клиентов).

-Товары: категория, модель, производитель, цены (в т.ч. средняя и минимальная), есть ли в наличии, описание, характеристики, внешний вид; магазины, где можно купить товар, их телефоны и адреса; аксессуары, их цены и где их купить.

- Магазины: название, компания-владелец, её юридический адрес и home-site, контактные телефоны, адрес, схема проезда, эмблема; товары и цены на них; рекламная информация: некоторые товары с фотографиями, описаниями и ценами, основные отделы (категории товаров).

9. 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.

10. 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.

11. 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).

12. 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.

13. Аналитический отдел некоторой компании (поиск и анализ публикаций).

- Категории: название, тип (область исследований, область приложений и т.п.), родительская категория, дочерние категории, связанные по смыслу категории (с пояснениями о связях), найденные публикации.
- Публикации: название, тип (газетная, книжная, web и т.п.), название, тип, адрес и телефон источника (газета, книга, сайт и т.п.), выходные данные (date-line), язык, реферат, ключевые слова, категории (с указанием степени уверенности отнесения к ним), текст и его тип (обычный, DOC, HTML, отсканированные картинки и т.п.), обзор.
- Задачи: тип задачи (классификация или поиск), сотрудник (создавший категорию или нашедший публикацию, ответственный за категорию или публикацию и т.п.), завершена ли работанад задачей.

14. Аналитический отдел некоторой компании.

- Организация: название, тип (промышленная, финансовая, торговая, исследовательская и т.п.), категория(и), организация-владелец (акционеры), страна, контактная информация; договорные отношения с другими организациями.
-Технология (продукт): название, категория(и), организация-разработчик и производитель(и), использующие организации.
- Человек: фамилия, имя, тип (начальник, менеджер, создатель технологии и т.п.), организация(и), в которой работает, контактная информация.

15. Компания по (разработке и) сопровождению программного обеспечения.

- Ошибка (bug): краткое и полное описание, срок поступления информации об ошибке, её источник (пользователь, тестировщик) и его координаты, уровень ошибки (критическая, важная, незначительная и т.п.), категория функциональности (интерфейс, данные, расчетный алгоритм, другое, неизвестная категория), часть проекта, модуль (пакет), программист, ответственный за модуль, программист, ответственный за исправление ошибки, срок исправления (необходимый и реальный), исправлена ли, проверено ли исправление тестировщиком.

Лабораторная работа № 3.
Тема: Построение базы данных в СУБД Access. Нормализация отношений.
Цель работы: изучить и закрепить на практике методы и средства СУБД по корректному заполнению и модификации таблиц БД и методы контроля вводимых данных путем связывания таблиц.
Теоретическая часть

Для создания новой таблицы необходимо открыть базу данных, перейти на вкладку Создать и выбрать желаемый пункт для создания таблицы.

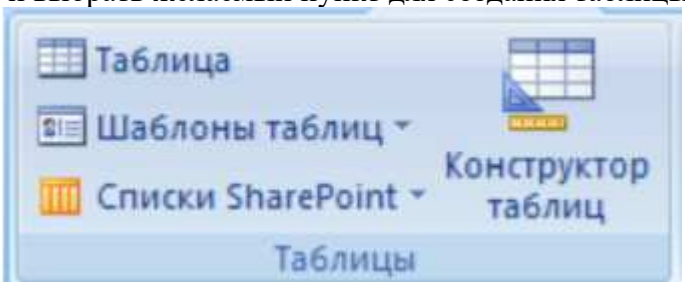


Рис.13. Окно выбора способа создания новой таблицы в СУБД MS Access В Access используются три способа создания таблиц: путем ввода данных (by entering data), с помощью Мастера создания таблиц (by using wizard) и с помощью Конструктора таблиц (in Design view), который является наиболее универсальным. В режиме Конструктора таблицы создаются путем задания имен полей, их типов и свойств. Чтобы создать таблицу в режиме Конструктора, необходимо:

1. Щелкнуть левой кнопкой мыши на Вкладке Создание Конструктор таблицы.

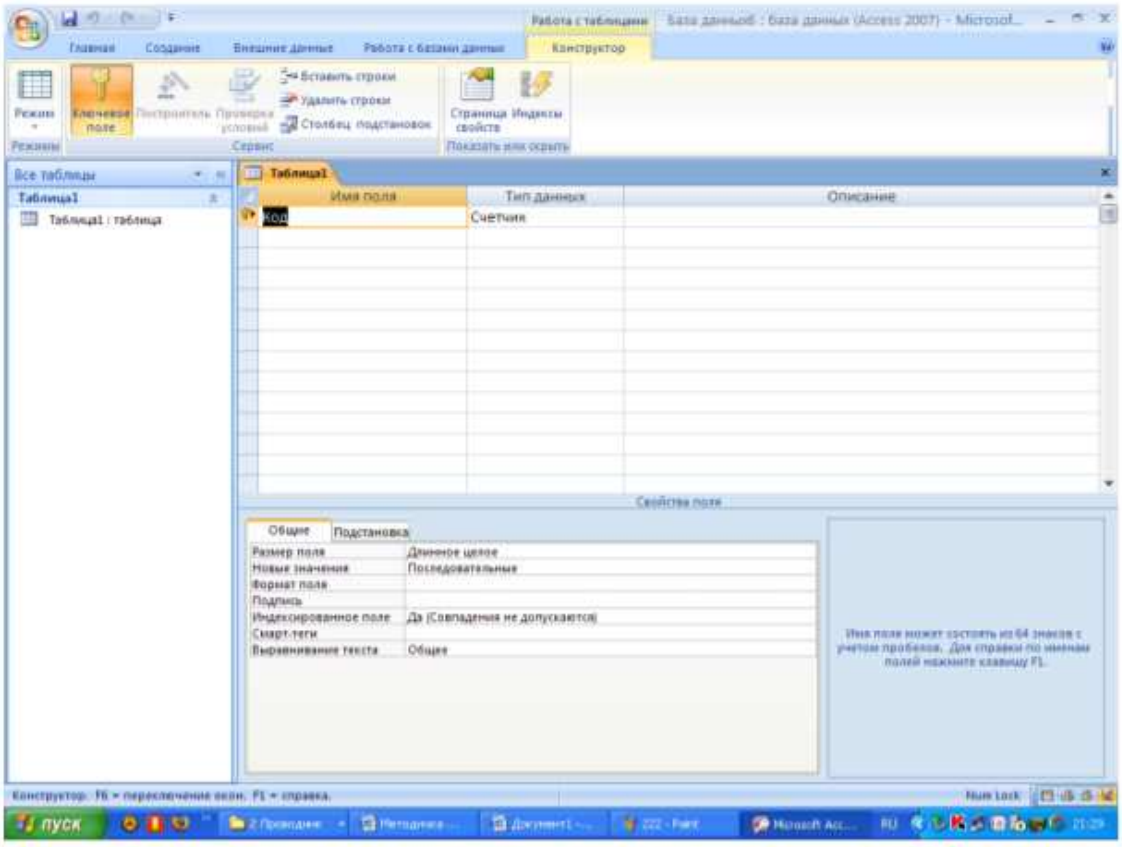


Рис.14. Окно создания новой таблицы в режиме Конструктора

2. В окне Конструктора таблиц в столбец Имя поля (Field Name) ввести имена полей создаваемой

таблицы.

3. В столбце Тип данных (Data Type) для каждого поля таблицы выбрать из раскрывающегося списка тип данных, которые будут содержаться вэтом поле.

4. В столбце Описание (Description) можно ввести описание данного поля (не обязательно).

5. В нижней части окна Конструктора таблиц на вкладках Общие (General) и Подстановка (Lookup) ввести свойства каждого поля или оставить значения свойств, установленные по умолчанию.

6. После описания всех полей будущей таблицы нажать кнопку Закрыть (в верхнем правом углу окна таблицы).

7. На вопрос Сохранить изменения макета или структуры таблицы <имя таблицы>? (Do you want to save changesto the design of table <имя таблицы>?), нажать кнопкуДа (Yes).

8. В окне Сохранитькак(SaveAs) в полеИмя таблицы(TableName) ввести имя создаваемой таблицы и нажать кнопку ОК.

9. В ответ на сообщение Ключевые поля не заданы (There is no primary key defined) и вопрос Создать ключевое поле сейчас? (Do you want to create a primary key now?) нажмите кнопку Да (Yes) если ключевое поле необходимо, или кнопку Нет (No) если такого не требуется.

10. После указанных действий в списке таблиц в окне базы данных появятся имя и значок новой таблицы. Ввести данные в созданную таблицу можно, открыв таблицу в режиме Таблицы.

В Microsoft Access имеются следующие типы данных:

-Текстовый (Text) — символьные или числовые данные, не требующие вычислений. Размер текстового поля задается с помощью свойства Размер поля (FieldSize), в котором указывается максимальное количество символов, которые могут быть введены в данное поле.

- Поле MEMO (MEMO) — поле MEMO предназначено для вводатекстовой информации, по объему превышающей 255 символов. Этот тип данных отличается от типа Текстовый (Text) тем, что в таблице хранятся не сами данные, а ссылки на блоки данных, хранящиеся отдельно. За счет этого ускоряется обработка таблиц (сортировка, поиск и т. п.).

-Числовой (Number) — числовой тип применяется для хранения числовых данных, используемых в математических расчетах. Дата/Время (Date/Time) — тип для представления даты и времени.

- Денежный (Currency) — тип данных, предназначенный для хранения данных, точность представления которых колеблется от 1 до 4 десятичных знаков.

- Счетчик (AutoNumber) — поле содержит 4-байтный уникальный номер, определяемый Microsoft Access автоматически для каждой новой записи либо случайным образом, либо путем увеличения предыдущего значения на 1. Значения полей типа счетчика обновлять нельзя.

- Логический (Yes/No) — логическое поле, которое может содержать только два значения, интерпретируемых как Да/Нет, Истина/Ложь, Включено/Выключено.

- Поле объекта OLE (OLE object) — содержит ссылку на OLE-объект (лист Microsoft Excel, документ Microsoft Word, звук, рисунок и т. п.). В поле объекта OLE могут храниться произвольные данные, в том числе и данные нескольких типов. Это позволяет обойти основное ограничение реляционных баз данных, которое требует, чтобы в каждом поле хранились данные только одного типа.

- Гиперссылка (Hyperlink) — дает возможность хранить в поле ссылку, с помощью которой можно ссылаться на произвольный фрагмент данных внутри файла или Web-страницы на том же компьютере, в Интернет.

При установке курсора на какое-либо из заполняемых полей таблицы в левой нижней части экрана появляются свойства этого поля. Причем свойства задаваемого поля зависят от принятого для него типа данных.

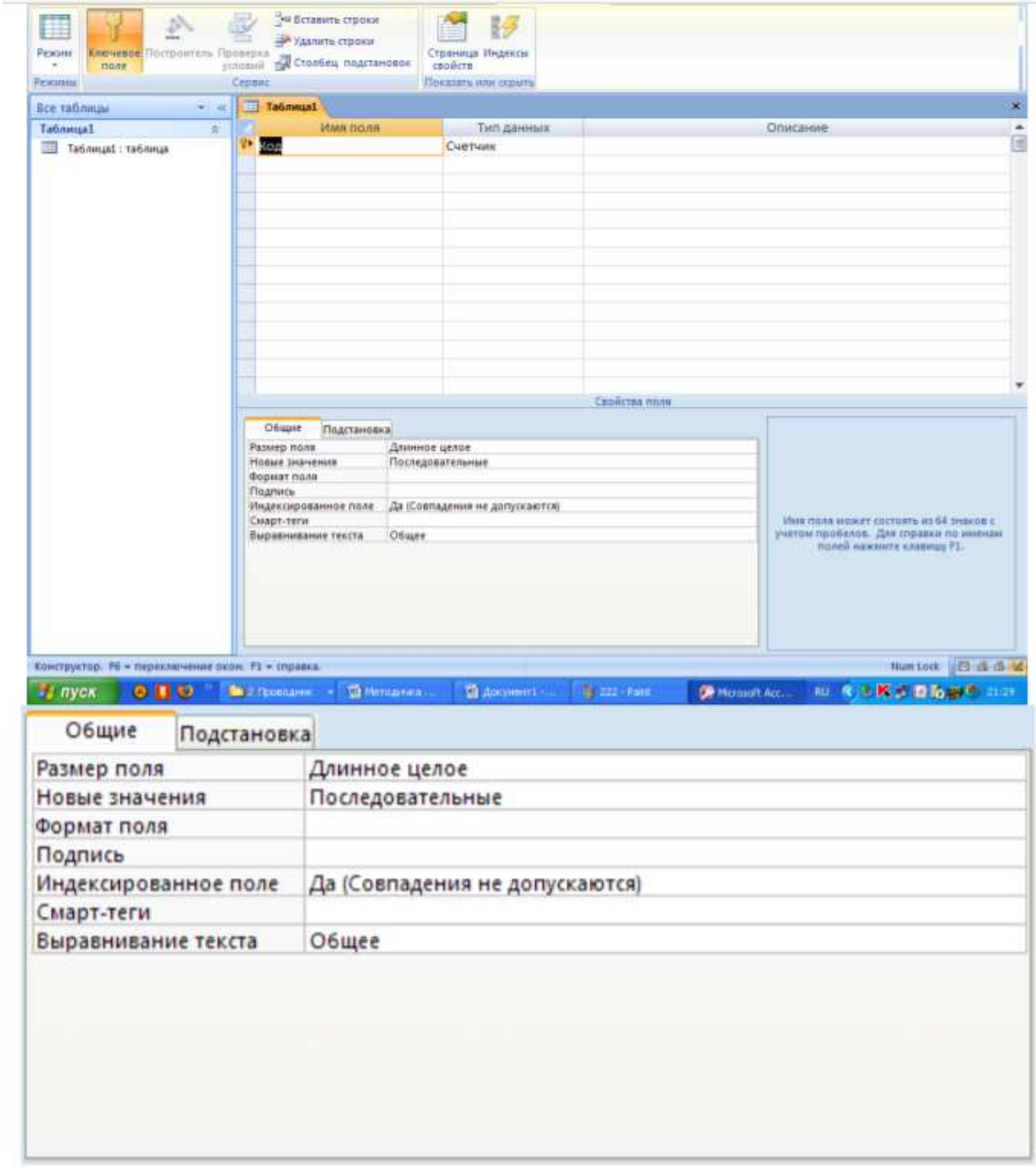


Рис.15. Установка свойств полей таблицы

Свойство Размер поля зависит от размера хранящейся в нем информации. Например, для ввода данных в поле и Фамилия работника достаточно 20 символов.

Для большинства типов данных характерно свойство поля - Подпись (Caption). С помощью этого свойства можно задать названия полей таблицы, которые выводятся в различных режимах (в надписях, присоединенных к элементам управления формы, в заголовке столбца в режиме Таблицы и т.д.). Кроме того, для большинства типов данных существует свойство Обязательное поле (Required), которое определяет необходимость ввода данных в это поле.

Свойство Формат поля (Format) указывает формат отображения данных из поля в режиме Таблицы. Для определения формата полей текстового типа используются специальные символы форматирования. Для числовых полей значение формата можно

выбрать из раскрывающегося списка. Для логических полей можно выбрать из списка следующие варианты: Да/Нет (Yes/No), Истина/Ложь (True/False), Вкл/Выкл (On/Off).

С помощью свойства Маска ввода (Input Mask) указывается маска, позволяющая автоматизировать проверку ввода символов в поле. Она применяется к таким полям, как номер телефона, дата и т. д. Задавать маску ввода можно вручную или с помощью Мастера.

Проще всего научиться работать с маской ввода с помощью мастера по созданию масок ввода. Для создания маски с помощью мастера необходимо щелкнуть на соответствующее поле, затем - по ячейке свойства Маска ввода, расположенной в нижней части этого окна. Вы увидите справа небольшую кнопку с тремя точками – кнопку построителя. Нажмите эту кнопку, чтобы воспользоваться помощью мастера по созданию масок ввода. Выберите один из предлагаемых стандартных масок и нажмите далее.

Свойство Индексированное поле (Indexed) определяет, является ли данное поле индексированным, и если является, то в каком режиме. Существуют два режима индексирования: Совпадения допускаются (Duplicates OK) и Совпадения не допускаются (No duplicates).В первом случае поле может содержать повторяющиеся значения, во втором— нет. Два свойства, которые тоже определены для большинства полей, позволяют выполнять проверку данных, вводимых в поле:

- Условие на значение (Validation Rule) — свойство определяет условие (ограничение), накладываемое на вводимые в это поле данные. При несоответствии вводимых данных указанному условию выдается сообщение об ошибке. Для создания условия на значение с помощью мастера необходимо щелкнуть на соответствующее поле, затем - по ячейке свойства Условие на значение, расположенной в нижней части этого окна. Вы увидите справа небольшую кнопку с тремя точками - кнопку построителя. Нажмите эту кнопку, на экране появится построитель выражения, с помощью которого вы сможете задать любое условие.

-Сообщение об ошибке (Validation Text) — свойство определяет то сообщение, которое будет выдаваться пользователю, если при вводе данных не соблюдается условие, указанное в свойстве Условие на значение (Validation Rule). Например, "Введенное значение выходит за рамки заданного «диапазона». Ввод различных выражений в Microsoft Access возможен не только вручную, но и может быть облегчен с помощью Построителя выражений. Построитель выражений вызывается всякий раз, когда в поле свойства объекта Access вы щелкните кнопку Построить  на строке главного меню или нажимаете кнопку Построителя (кнопка с тремя точками, например, в строке свойства Условие на значение).

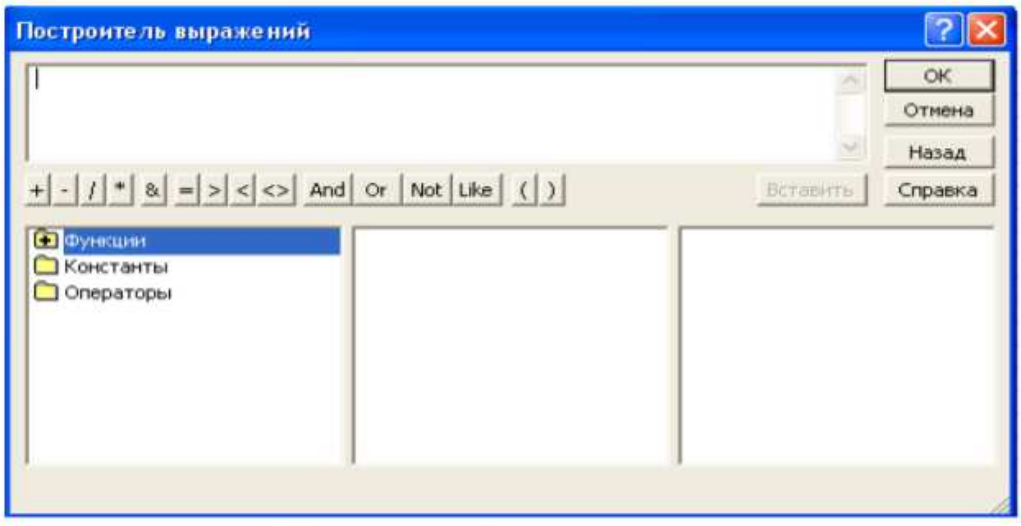


Рис.16. Окно Построителя выражений

С помощью удобных клавиш, расположенных в Построителе можно задавать любое выражение, в том числе использовать различные формулы для вычислений и применять встроенные функции Access. Введение выражений с помощью Построителя позволяет также наглядно определить правильность ввода и тем самым избежать неточностей и ошибок при наборе текста и формул. Создать поле-список и сделать более простым ввод значений в поле таблицы позволяет операция подстановки. Используя эту операцию, при последующем заполнении таблицы данными можно выбирать значения поля из списка. Список значений может быть как фиксированным, так и содержаться в таблице или запросе. Сформировать столбец подстановок для поля помогает Мастер подстановок (Lookup Wizard), который находится в поле Тип данных. Проверить наличие подстановки можно, открыв вкладку Подстановка, в свойствах поля. Выше неоднократно упоминалось понятие ключевого поля. Ключевое поле — это одно или несколько полей, комбинация значений которых однозначно определяет каждую запись в таблице. Если для таблицы определены ключевые поля, то Microsoft Access предотвращает дублирование или ввод пустых значений в ключевое поле.

Ключевые поля используются для быстрого поиска и связи данных из разных таблиц при помощи запросов, форм и отчетов.

Операция сортировки данных используется всегда для удобства нахождения нужной информации. Когда на экране (или на бумаге) отображается таблица, гораздо легче найти нужную строку, если эти строки упорядочены. Вы привыкли к тому, что табличные данные упорядочены по алфавиту, по дате, по увеличению или уменьшению значений в столбцах, содержащих числа. Но в разных ситуациях мы хотели бы сортировать строки по разным признакам (столбцам таблицы). В идеале это должно выполняться легким движением руки. Именно так и позволяет делать Access. По умолчанию, когда таблица открывается в режиме Таблицы, она упорядочивается по значению ключевого поля. Если ключевое поле для таблицы не определено, записи выводятся в порядке их ввода в таблицу. Если нужно отсортировать записи по значению другого поля, достаточно установить курсор на любую строку соответствующего столбца и нажать одну из кнопок на панели инструментов: Сортировка по возрастанию или Сортировка по убыванию. Создание Связей и индексов:
Каждая БД представляет обычно несколько таблиц, число которых может доходить, в общем случае, до десятков и сотен. При этом часто оказывается, что в разных таблицах хранятся одинаковые данные. Для связывания полей необязательно совпадение их имен, но обязательно совпадение их типов.

Связи между таблицами можно устанавливать двумя путями. Первый путь - графический. Войдите в Схему данных, и выберите таблицы для установления связей. Далее следует выбрать в главной таблице поле для связи, нажать левую кнопку мыши и перетащить поле во вторую таблицу. Отпустить левую кнопку мыши над тем полем подчиненной таблицы, с которым устанавливается связь. Второй путь - создание связей через Мастер подстановок, который активизируется при выборе типа данных в конструкторе таблицы.

Чтобы изменить связь нужно войти в Схему данных (ярлык на главной панели инструментов), щелкнуть правой кнопкой мыши по линии связи и войти в диалоговое окно Изменить связь, установив обеспечение целостности данных, каскадное обновление и каскадное удаление записей связанных таблиц. Если отношение между таблицами «один-ко-многим», то слева из списка Таблица/запрос выбирается главная таблица и поле в этой таблице, а справа из списка Связанная таблица/запрос - подчиненная и соответственно поле в ней. Если отношение «один-к-одному», то порядок таблиц значения не имеет. Если вы уже устанавливали связь графически, то все поля в списке уже выбраны, и нужно только определить правила ссылочной целостности. Для этого устанавливают флажок Обеспечение целостности данных. Когда создается новая связь, можно также воспользоваться кнопкой Новое и в окне Создание ввести имена связываемых таблиц и имена полей, используемых для связи. Нажать кнопку ОК. Индексы - это внутренняя таблица, состоящая из двух столбцов: значения выражения, содержащего все поля, включенные в индекс, и местоположение каждой записи таблицы с данным значением индексного выражения.

Допустим, вы часто осуществляете поиск Сотрудников в таблице Сотрудники по Фамилии, Имени и Отчеству. Если индекс отсутствует, то Access просматривает все записи вашей таблицы. Эта операция выполняется быстро только при небольшом количестве записей. Если вы создадите индекс, то Access его использует для прямого поиска записей. Индексы бывают по одному полю и составные. Для создания индекса по одному полю откройте таблицу в режиме Конструктора и откройте поле, для которого вы хотите создать индекс. Щелкните по ячейке Свойства Индексированное поле, расположенное в нижней части окна таблицы и установите значение Да. Если поле, по которому вы индексируете имеет повторяющиеся значения, то следует выбрать Да (совпадения допускаются). Однако Access также может создать индекс, содержащий только уникальные значения данного поля - для этого нужно выбрать в списке значений Да (совпадения не допускаются). Access автоматически создает такой индекс по первичному ключу таблицы. Чтобы создать индекс по нескольким полям, откройте таблицу Сотрудники в режиме Конструктора, затем нажмите кнопку Индексы на панели инструментов в режиме Конструктора.

Рис. 17. Диалоговое окно создания Индекса

В столбце Индекс заполняется название вашего индекса, в столбце Имя поля указывается наименование индексируемого поля, в столбце Порядок сортировки можно указать, как вы хотите отсортировать данные, по возрастанию или по убыванию.

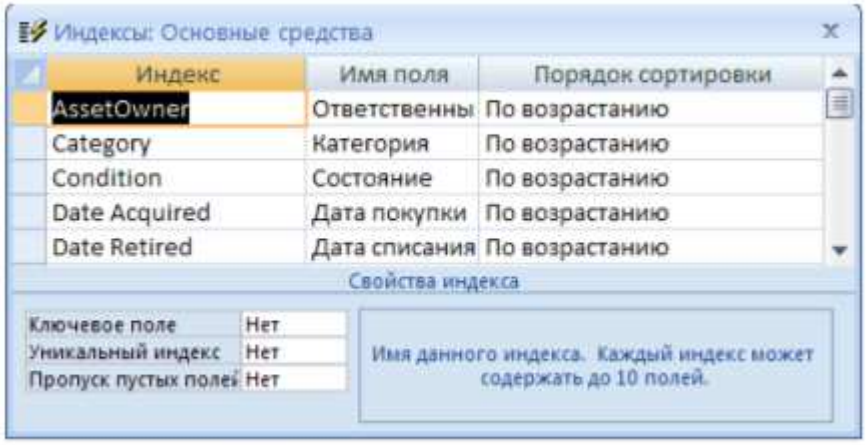


Рис. 18. Создание нескольких индексов

Для построения составного индекса поместите курсор на пустую строку и введите с клавиатуры имя индекса. В столбце Имя поля окна индексы выберите необходимое поле, пользуясь раскрывающимся списком, затем, не указывая никакого нового названия индекса, поместите курсор на следующую строку столбца Имя поля и укажите следующее поле, которое будет включено в составной индекс.

Задание. 1.

1.Разработать структуру базы данных согласно вариантам задания.

2. Создать таблицы в среде Microsoft Access. Для каждого элемента данных определить имя, тип данных, свойства: размер, условия на значение, маску ввода, сообщение об ошибке, значение по умолчанию и т.д. Определить для таблиц первичные ключи.

3. Организовать связь между таблицами.

4. Заполнить таблицы данными.

Лабораторная работа № 4.

Тема: Создание запросов.

Цель работы: изучение и закрепление на практике методов формирования и использования запросов для выборки данных в таблицах.

Теоретическая часть

Таблицы только хранят данные, но необходимо иметь возможность выбрать заданные данные из нескольких таблиц. Именно для этого служат запросы на выборку. В запросах на выборку данные могут: отбираться по многим критериям; сортироваться; с ними могут производиться вычислительные операции. Запрос это временная таблица. Это значит, что данные в них не хранятся постоянно, а только временно вызываются из таблиц, по заранее заданному шаблону, в момент активизации запроса. Таким образом, в базе данных постоянно хранится только шаблоны вызова данных (временные таблицы удаляются после закрытия запроса), а сама информация не дублируется.

Запросы позволяют:

1. формировать сложные критерии для выбора записей из одной или нескольких таблиц;
2. указывать поля, которые должны быть отображены для выбранных записей;
3. редактировать группы записей, удовлетворяющих определенным критериям;
4. выполнять вычисления с использованием выбранных данных.

Для создания запроса необходимо:

1. Открыть свою базу данных.
2. Перейти на вкладку Запросы
3. Во вкладке Создание выбрать команду Конструктор запросов. После нажатия этой опции появится окно конструктора запроса с диалоговым окном добавления таблиц. Окно добавления таблиц можно также вызвать командой Добавить таблицу из меню Запрос.

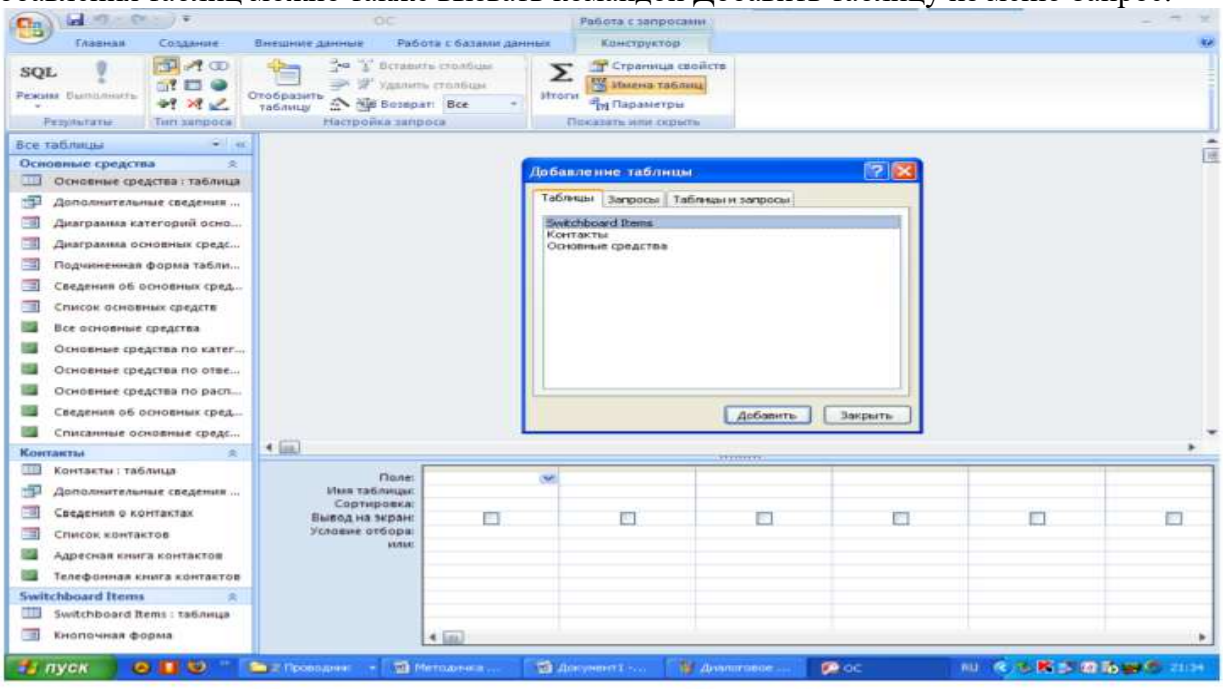


Рис. 19. Окно создания запроса в режиме Конструктора

4. Добавить в запрос необходимые таблицы
5. Убедиться, что между добавленными таблицами установлены связи.
6. После добавления таблиц нажать кнопку Закрыть в окне Добавление таблицы.

7. Затем нужно указать, какие поля из базовых таблиц будут отображаться в запросе. Для этого, выделить нужное поле в таблице-источнике, подвести указатель мыши к выделенному полю, нажать на левую кнопку мыши и перетащить поле в нужное место бланка запроса.

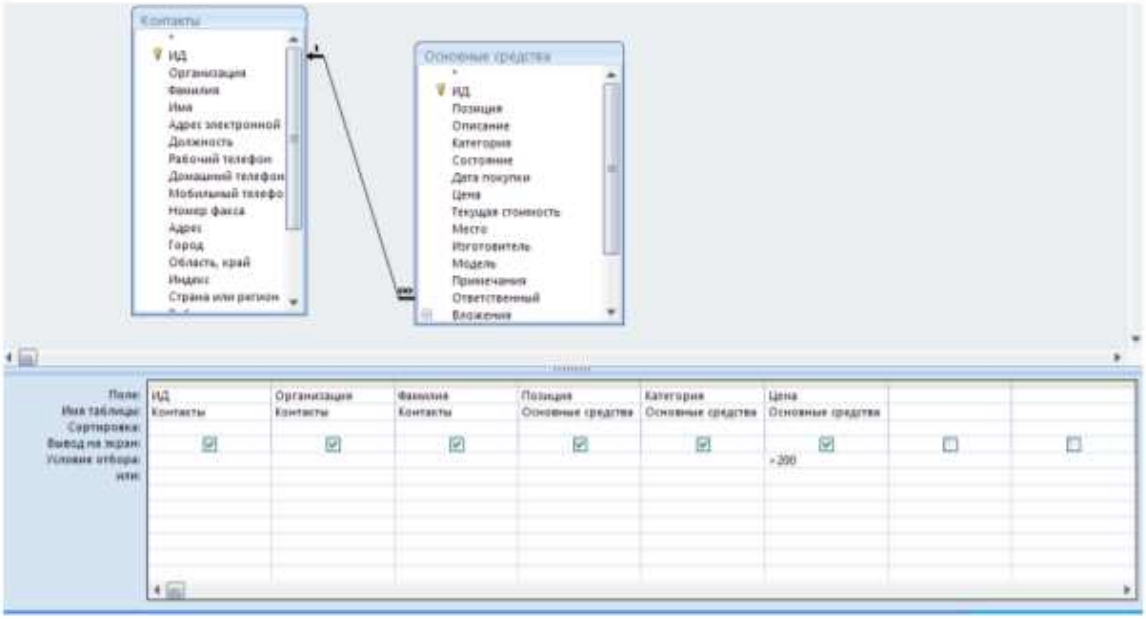


Рис. 20. Окно формирования запроса на выборку

8. В строке Условие отбора указывается необходимые условия для запроса. Например, призадании отобрать товары в ценовой категории Больше 200 в Условии отбора ставится >200

9. Сохранение запроса для дальнейшего использования производится нажатием на панели инструментов кнопки Сохранить. Далее СУБД запросит имя сохраняемого запроса. Целесообразно, чтобы оно имело смысловую нагрузку, что облегчит дальнейшее использование запроса. Чтобы удалить лишнюю или внесенную по ошибке базовую таблицу из запроса, необходимо выделить ее, щелкнув на любом месте в списке ее полей, и нажать клавишу Delete. Чтобы удалить поле из запроса, выделите нужный столбец в бланке запроса, а затем нажмите клавишу Delete. Самое главное в запросе - возможность использования критериев выборки, которые вводятся в строку Условие отбора.

Можно выделить следующие типы запросов на основе критериев выборки:

- Выборка по строгому совпадению. В строку Условие отбора для определенного поля вводится одно из значений, существующих в таблице. Например, название конкретного товара или название фирмы, города. Данные запросы можно параметризовать, т.е. вводить условия отбора в виде параметра при каждом запуске запроса, что устраняет необходимость предварительно его модификации. Для параметризации необходимо в строке Условие отбора вместо самого условия ввести текст приглашения на его ввод по формату [текст приглашения]. Например, [Введите наименование организации].

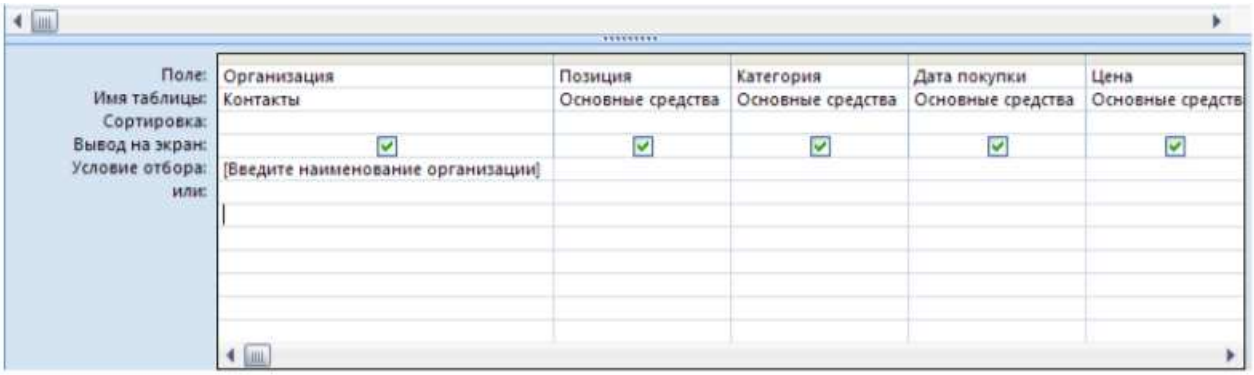


Рис. 21. Окно создания параметризированного запроса

Затем запрос сохраняется обычным способом. Чтобы оценить результаты сформированного запроса, необходимо его запустить двойным щелчком мыши. При запуске параметризованного запроса появляется диалоговое окно (рис. 2.4), в котором пользователь должен ввести собственно условие отбора и нажать клавишу ОК.

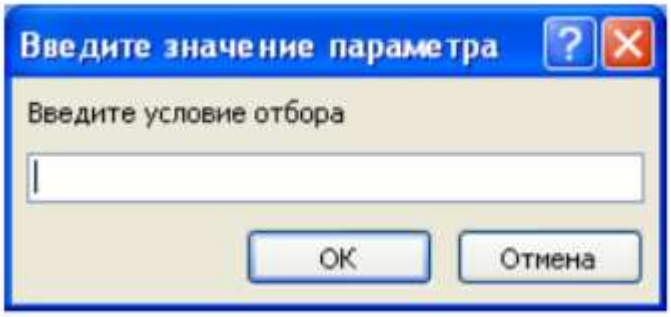


Рис. 22. Диалоговое окно запуска параметризированного запроса

-Выборка по строгому несовпадению. В этом случае в выборку отбираются все записи таблицы, кроме записей, содержащих значение, указанное в строке Условие отбора. Для реализации данного запроса перед значением вводится префикс Not или <>. Например, Not "МТФ" в поле Факультет запроса к таблице СТУДЕНТ приведет к выборке всех студентов вуза, кроме студентовМТФ.

- Выборка по неточному совпадению. Для выборки записей в условиях неполноты знаний о требуемых значениях используется оператор Like . В списке можно указывать сразу диапазон символов. Например, Like "[В-М]*" - выбираются все студенты, ФИО которых начинается на буквы от "В" до "М". Длина ФИО произвольная.

- Выборка по диапазону. Для формирования данных условий выбора используются операторы сравнения >, >=, <= и <>. Операции сравнения могут связываться логическими операциями And (И) и Or (ИЛИ). Для этих же целей используется оператор диапазона Between and . Например, выбор книг стоимостью от 100 до 200 рублей может быть реализован через ввод в запросе условия в поле Стоимость в виде >=100 and <=200 или Between100 and 200. Перечень значений в условии выборки можно задать и оператором In (значение, значение, ...). Например, выбор студентов факультетов МТФ или ФАПУ можно реализовать, указав в поле Факультет запроса условие In ("МТФ", "ФАПУ"). Это же условие можно записать и через операцию ИЛИ: "МТФ" or "ФАПУ". Также можно указать одно название факультета в строке Условие отбора (см. рис. 2.2), а второе в следующей строке или. Число строк или не ограничено. В выражениях отбора также можно использовать знаки математических операций +, -, /, * и неограниченное число круглых скобок. Сложные выражения в условиях отбора могут формироваться с помощью соответствующего построителя, который вызывается кнопкой на панели инструментов.

- Запрос с вычислениями. Такой запрос позволяет получить дополнительную информацию в процессе выборки, например, стоимость всей партии товара при хранимой в таблице

информации о количестве товара и стоимости единицы его продукции. Для этого в строку Поле пустого столбца заносят выражение для вычисления по следующему формату: :. Например: Стоимость партии:[Товар]![количество товара]*[стоимость единицы товара].

-Запрос с групповыми операциями. Рассмотренные запросы анализируют отдельные записи таблицы. Вместе с тем, СУБД Access позволяет находить интегральные показатели для групп записей в таблице. Каждая такая группа характеризуется одинаковым значением по какому-то полю, например, одинаковым названием факультета или семейным положением. Для перехода в данный режим запросов необходимо в панели инструментов нажать клавишу Групповые операции , что приведет к появлению в бланке запроса новой второй строки с одноименным названием. В ячейках данной строки указывается или режим группировки по некоторому полю (опция Группировка), или название групповой операции: * Sum - сумма значений * Avg - среднее значение по данному полю для всей группы; * Count - число записей в данной группе; * Max - максимальное значение поля в каждой группе; * Min - минимальное значение поля в каждой группе; * First - первое значение данного поля в каждой группе; * Last - последнее значение данного поля в каждой группе и др.

Опции выбора вызываются нажатием кнопки раскрытия в требуемой ячейке. При запуске запроса СУБД разбивает таблицу на группы, число которых равно числу существующих значений в группируемом поле, и реализует для каждой группы требуемую операцию, т.е. число строк в выборке равно числу групп. Рассмотренные запросы не изменяют содержимое исходной таблицы. Для реализации подобных действий СУБД Access использует четыре следующих запроса:

- Запрос-создание новой таблицы. Предназначен для сохранения результатов запроса в виде новой таблицы. Исходно формируется обычный запрос на выборку необходимой информации из таблицы. После проверки результатов его выполнения производится возврат в режим конструктора запросов. Далее нажимается кнопка Тип запроса на панели инструментов или выбирается команда главногоменю Запрос. В появившемся списке выбирается опция Создание таблицы, после чего СУБД запрашивает её имя. Указывается имя создаваемой таблицы и нажимается кнопка ОК. Непосредственно запрос на создание запускается нажатием кнопки на панели инструментов. В окне Таблицы БД появляется пиктограмма созданной таблицы.
- Запрос-добавление выборки в другую таблицу. Выборку можно добавить к другой таблице, однотипной по структуре или с изменением структуры выборки. Для этого необходимо сформировать обычный запрос и оценить результаты его выполнения. Далее следует вернуться в режим конструктора запроса. Нажимается кнопка Тип запроса на панели инструментов или выбирается команда главного меню Запрос. В появившемся списке выбирается опция Добавление, после чего СУБД запрашивает имя таблицы, к которой будет добавлена выборка. Последний шаг - нажатие кнопки ОК. Выборку можно добавлять и к таблицам других БД, что определяется установкой соответствующих переключателей в окне ввода имени целевой таблицы. Если структура выборки и целевой таблицы не совпадают, то в целевую таблицу добавляются значения только тех полей выборки, имена которых совпадают с именами полей целевой таблицы.
- Запрос-удаление. С помощью запросов можно удалить часть или все записи из таблицы. Для этого необходимо сформировать обычный запрос и оценить результаты его выполнения. Далее следует вернуться в режим конструктора запроса. Нажимается кнопка Тип запроса на панели инструментов или выбирается команда главного меню Запрос. В появившемся списке выбирается опция Удаление, после чего в бланке запроса появляется новая третья строка с именем Удаление, куда можно вводить дополнительные условия на выборку удаляемых записей. Последний шаг - нажатие кнопки ОК.
- Запрос-обновление. С помощью запросов можно обновлять в единой операции некоторые или все значения выбранных полей. Для этого необходимо сформировать обычный запрос и оценить результаты его выполнения. Далее следует вернуться в режим конструктора запроса.

Нажимается кнопка Тип запроса  на панели инструментов или выбирается команда главного меню Запрос. В появившемся списке выбирается опция Обновление, после чего в бланке запроса появляется новая третья строка с именем Обновление. В ней задаются новые значения полей таблицы, в том числе и вычисляемые выражения. Далее запрос запускается на выполнение кнопкой. СУБД указывает число модифицируемых записей и просит подтвердить изменения кнопкой ОК. Пользователь на этом этапе еще может отказаться от модификации значений в таблице. В предыдущих лабораторных работах мы уже использовали так называемый Построитель выражений. Его используют для построения сложных выражений в Access. Так, если вам в работе потребуется применить встроенную функцию, щелкните в построителе выражений по папке Функции в самом левом списке, чтобы вывести категории функции. Например, в одном из заданий вам необходимо вывести в поле запроса Фамилию +инициалы. Для этого следует использовать встроенную функцию Left текстового типа. Затем вам необходимо набрать выражение ФИО: [Сотрудники]![ФАМИЛИЯ]+"пробел"+Left([Сотрудники]![Имя];1)+"точка"+Left([Сотрудник и]![Отч ество] ;1)+"точка" В кавычках ставится реальный пробел и реальная точка. Таким образом, в вашем запросе будут отражаться Иванов И.И

пасности.

Задание.

Для всех вариантов задания 1 сформировать и выполнить следующие запросы:

1. Запрос-выборку по одной таблице с использованием параметров.
2. Запрос-выборку по нескольким таблицам и запросам с использование вычисляемых полей.
3. Запрос на обновление данных.
4. Запрос на удаление данных.
5. Запрос на создание таблицы.
6. Итоговый запрос.
7. Перекрестный запрос.

Предложенные ниже варианты запросов попытаться выполнить с помощью запросов по образцу QBE, в противном случае реализовать SQL-запросы.

Варианты заданий

Вариант 1.

1. Определить покупателя, который купил максимальное количество товаров.
2. Для каждой покупки рассчитать общую стоимость.
3. Определить сумму продажи для каждого месяца.
4. Определить покупателей, купивших товаров на сумму, превышающую среднюю сумму покупок всех покупателей.
5. Определить тип, товаров которого куплено больше всего.

Вариант 2.

1. Для каждого вида изделия рассчитать его стоимость.
2. Найти изделия, в состав которых входит больше всего компонентов.
3. Определить компоненты, которые входят в большее число изделий.
4. Вычислить прибыль от продажи каждого типа продукции.
5. Найти изделие, на сборку которого уходит дней больше, чем в среднем на сборку изделий.

Вариант 3.

1. Определить заказ, на выполнение которого ушло больше всего дней.
2. Определить клиентов, стоимость заказов которых превысила их кредит.
3. Рассчитать стоимость каждого заказа с учетом скидки.
4. Определить клиента, который купил больше всего товаров.
5. Определить город, где живет клиент, чаще других оформляющий заказы.

Вариант 4.

1. Рассчитать для каждого квартиросъемщика квартплату.

2. Определить задолжников по квартплате за каждый месяц.

3. Определить дом с максимальной жилой площадью.

4. Определить дом с максимальной плотностью населения.

5. Жильцам, просрочившим оплату жилья, назначить пени 1% за каждый просроченный

день.

Вариант 5.

1. Рассчитать общую стоимость товара с учетом транспортных расходов, скидки и налога.

2. Определить прибыль от продажи за каждый месяц

3. Определить страну, в которой изготовлены компоненты, вошедшие в товар, пользующийся наибольшей популярностью.

4. Определить самый дешевый компонент, поступающий без лицензии.

5. Определить товар, в состав которого входят компоненты с максимальными транспортными расходами.

Вариант 6.

1. Вычислить оплату труда каждого сотрудника как нижняя грань оплаты, если сотрудник работает меньше года, средняя оплата, если сотрудник работает от года до пяти лет, и верхняя грань - более пяти лет.

2. Для каждого сотрудника рассчитать его ежемесячный заработок.

3. Найти сотрудника, который работает дольше других.

4. Вычислить сумму налога, которую фирма платить каждый месяц.

5. Определить сотрудников, ежемесячная оплата которых оказалась больше средней.

Вариант 7.

1. Определить услугу, пользующую наибольшей популярностью.

2. Для каждого клиента рассчитать стоимость услуг с учетом социального положения и скидок.

3. Определить доход фирмы от предоставленных услуг за каждый месяц.

4. Определить, жители города или села чаще всего обращаются в фирму.

5. Рассчитать количество и сумму предоставленных населению услуг по категориям, определенным социальным происхождением клиентов.

Вариант 8.

1. Для каждого клиента вычислить сумму оплаты междугородних разговоров.

2. Определить город, с которым чаще всего разговаривают клиенты.

3. Определить клиента, который говорит по телефону чаще и дольше других.

4. Определить время суток, на которое приходится больше всего разговоров.

5. Определить день, в который телефонная линия была занята меньше всего.

Вариант 9.

1. Вычислить стоимость и калорийность каждого блюда.

2. Определить блюдо из супов с наименьшим содержанием жиров.

3. Определить компоненты самого дорогого блюда.

4. Найти компонент, который входит в большинство блюд.

5. Определить содержание жира, белков и углеводов в самом дорогом блюде самого дешевого в среднем типа блюд.

Вариант 10.

1. Определить тематику, по которой продается больше всего книг.

2. По каждому месяцу вычислить сумму продаж.

3. Определить, книги каких авторов пользуются наибольшей популярностью, авторов мужчин или авторов-женщин.

4. Определить дни, когда было продано книг больше, чем обычно (т.е. больше среднего).

5. Какие по тематике книги пишут молодые авторы.

Вариант 11.

1. Вычислить сумму продаж по каждому месяцу.
2. Определить страну, выпустившую самый долгозвучащий диск.
3. Определить песню, пользующуюся наибольшей популярностью.
4. Составить рейтинг исполнителей по каждому месяцу.
5. Определить автора слов, написавшего больше всех песен.

Вариант 12.

1. Определить сумму продаж по каждому месяцу.
2. Определить, какой тип покупателей чаще других покупает видеокассеты.
3. Какой самый старый фильм был продан за последний месяц.
4. Определить страну, завоевавшую своими фильмами больше сего Оскаров.
5. Какие по тематике фильмы смотрит молодежь.

Вариант 13.

1. Для каждой олимпиады рассчитать отношения числа завоеванных медалей к числу участников.
2. Определить команду, для которой отношение числа завоеванных золотых медалей к числу участников больше, чем аналогичный показатель олимпиады.
3. Определить команды, которые чаще других участвовали в олимпиадах.
4. Определить команды, которые по числу завоеванных медалей на протяжении всех олимпиад попадали в первую тройку.
5. Найти олимпиады, символы которых совпадали.

Вариант 14.

1. Определить предмет, по которому нет двоек.
2. Определить студентов, сдавших успешно экзамены и набравших в сумме часов больше, чем среднее число часов по всем студентам.
3. Определить блок дисциплин, средняя оценка по которым самая высокая.
4. Определить кафедру, по предметам которой получено больше всего двоек студентами младших курсов.
5. Найти студентов, сдавших все экзамены успешно, если их день рождения пришелся на период сдачи экзаменов.

Вариант 15.

1. Вычислить зарплату каждого преподавателя.
2. Определить блок дисциплин, которые читают самые квалифицированные преподаватели.
3. Определить кафедру, для которой отношение числа предметов к числу преподавателей самое большое.
4. Определить семестр и кафедры, на которые приходится учебная нагрузка в часах, больше средней по кафедрам.
5. Определить предмет, который читается в большинстве семестров.

Вариант 16.

1. Вычислить прибыль от каждого рейса.
2. Определить рейсы до заданного пункта, на которые остались свободные места.
3. Найти отношение количества рейсов дальнего следования, которые выполняют квалифицированные командиры экипажей, к общему числу рейсов дальнего следования.
4. Какой экипаж имеет больше всего налетов , по количеству и по продолжительности.
5. Какие пассажиры по своему социальному положению летают чаще других.

Вариант 17.

1. Определить водителей с плохим техническим состоянием автобуса6 чаще других отправляющихся в рейс.

2. Вычислить прибыль от поездок за каждый месяц.

3. Какие поездки пользуются наибольшей популярностью.

4. Для каждой организации рассчитать долг с учетом стоимости поездок и выплаченного аванса.

5. Определить водителя, совершившего больше всего поездок.

Вариант 18.

1. Для каждого вуза рассчитать объем свободных наличных средств.

2. Определить факультеты с самым большим отношением числа студентов к числу преподавателей.

3. Определить вуз с самой низкой средней стоимостью обучения одного студента.

4. Найти вуз с самым большим числом хоздоговорных студентов.

5. Определить, какая сумма приходится на каждого студента.

Вариант 19.

1. Определить, преступления какого вида раскрываются быстрее других.

2. Оружие какой страны наиболее часто используется в преступлениях.

3. Определить сколько преступлений и какого вида приходится на каждую возрастную группу.

4. В каком городе преступления раскрываются быстрее, чем в других городах.

5. В какой месяц было совершено больше всего преступлений.

Вариант 20.

1. В каком месяце была продана самая дорогая из старых машин.

2. Машины какой страны пользуются популярностью у молодежи.

3. Определить сумму продаж за каждый месяц.

4. Определить, какая возрастная группа покупает в среднем самые дорогие автомобили.

5. Какая группа по социальному положению предпочитает при расчете кредитные карточки.

Лабораторная работа № 5.

Тема: Введение ограничений целостности базы данных в СУБД Access

Цель работы: Изучение и закрепление на практике методов обеспечения целостности данных в реляционных базах данных.

Теоретическая часть

Целью обеспечения целостности данных является предотвращение появления непарных записей, ссылающихся на несуществующие записи. Обеспечение целостности данных включается для конкретного отношения между таблицами. В результате Access отменяет для этого отношения все действия, которые могут нарушить целостность данных. Это означает, что будет отменено как обновление, изменяющее целевой объект ссылки, так и удаление такого целевого объекта. Сведения о том, как настроить в Access распространение операций обновления и удаления таким образом, чтобы в результате изменялись и все связанные строки.

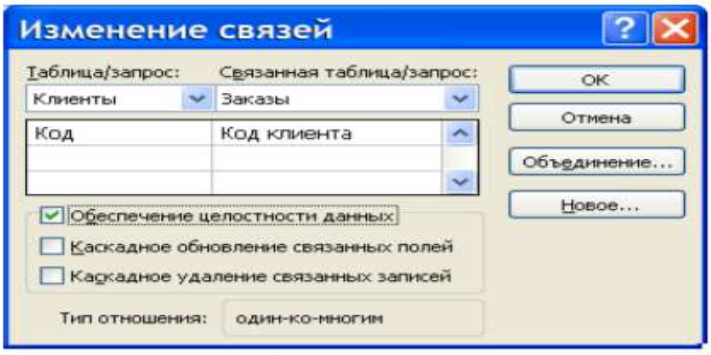


Рис.23. Диалоговое окно **Изменение связей**.

Включение и отключение обеспечения целостности данных

1. На вкладке Работа с базами данных в группе Отношения нажмите кнопку Схема данных.

2.На вкладке Конструктор в группе Связи нажмите кнопку Все связи. Будут отображены все таблицы с отношениями, а также соответствующие линии связи. Обратите внимание на то, что скрытые таблицы (таблицы, для которых установлен флажок скрытый в диалоговом окне Свойства) и их отношения не отображаются, если в диалоговом окне Параметры переходовне выбран параметр «Показывать скрытые объекты».

3.Щелкните линию отношения, которое вы хотите изменить. При выделении линия связи становится толще.

4. Дважды щелкните линию связи. Откроется диалоговое окно Изменение связей.

5. Выберите или отмените параметр Обеспечение целостности данных.

6. Внесите в отношение необходимые изменения и нажмите кнопку ОК. В режиме обеспечения целостности данных действуют перечисленные ниже правила.

- Не допускается ввод в поле внешнего ключа связанной таблицы значений, отсутствующих в поле первичного ключа главной таблицы, поскольку это приводит к появлению потерянных записей.

- Не допускается удаление записи из главной таблицы, если в связанной таблице существуют связанные с ней записи. Например, невозможно удалить запись из таблицы "Сотрудники", если в таблице «Заказы» имеются заказы, относящиеся к данному сотруднику. Однако можно удалить главную запись и все связанные записи одним действием, установив флажок Каскадное удаление связанных записей.

-Не допускается изменение значения первичного ключа в главной таблице, если это приведет к появлению потерянных записей. Например, нельзя изменить номер заказа в таблице «Заказы», если в таблице «Сведения о заказах» имеются строки, относящиеся к этому заказу. Однако можно обновить главную запись и все связанные записи одним действием, установив флажок «Каскадное обновление связанных полей»

Примечания: Если при включении обеспечения целостности данных возникли трудности, обратите внимание на то, что должны выполняться условия, перечисленные ниже.

- Общее поле главной таблицы должно быть первичным ключом или иметь уникальный индекс.
- Общие поля должны иметь одинаковый тип данных. Единственное исключение — поле типа «Счетчик» можно связать с полем типа «Числовой», если его свойство Размер поля имеет значение Длинное целое.
- Обе таблицы существуют в одной базе данных Access. Обеспечение целостности данных нельзя включить для присоединенных таблиц. Однако если исходные таблицы имеют формат Access, можно открыть базу данных, в которой они хранятся, и включить обеспечение целостности данных в этой базе.

Задание каскадных параметров

Иногда возникает ситуация, в которой требуется изменить значение только на стороне "один" отношения. В этом случае необходимо, чтобы Access автоматически обновил все затронутые строки в ходе выполнения одной операции. Тогда обновление будет полностью завершено, а база данных не будет находиться в несогласованном состоянии, когда некоторые строки обновлены, а другие — нет. Этой проблемы можно избежать с помощью параметра Access "Каскадное обновление связанных полей». Если при включении обеспечения целостности данных был включен параметр «Каскадное обновление связанных полей», то при последующем обновлении первичного ключа автоматически будут обновлены все связанные с ним поля.

Также может потребоваться удаление строки и всех связанных с ней записей — например, записи в таблице «Поставщики» и всех связанных с этим поставщиком заказов. Для этого в Access предназначен параметр «Каскадное удаление связанных записей». Если

включить обеспечение целостности данных и установить флажок Каскадное удаление связанных записей, при удалении записи, содержащей первичный ключ, будут автоматически удалены все записи, связанные с этим первичным ключом.

Включение и отключение каскадного обновления и каскадного удаления

1. На вкладке Работа с базами данных в группе Отношения нажмите кнопку Схема данных.
2. На вкладке Конструктор в группе Связи нажмите кнопку Все связи. Будут отображены все таблицы с отношениями, а также соответствующие линии связи. Обратите внимание на то, что скрытые таблицы (таблицы, для которых установлен флажок скрытый в диалоговом окне Свойства) и их отношения не отображаются, если в диалоговом окне Параметры переходов не выбран параметр «Показывать скрытые объекты».

3.Щелкните линию отношения, которое вы хотите изменить. При выделении линия связи становится толще.

4. Дважды щелкните линию связи. Откроется диалоговое окно Изменение связей.

5. Установите флажок Обеспечение целостности данных.

6.Установите флажок Каскадное обновление связанных полей, Каскадное удаление связанных записей или оба эти флажка.

Внесите в отношение необходимые изменения и нажмите кнопку ОК.

Задание.

Примечания: Если при включении обеспечения целостности данных возникли трудности, обратите внимание на то, что должны выполняться условия, перечисленные ниже.

1. С помощью механизма каскадных изменений обеспечить поддержку ссылочной целостности Вашей базы данных.
2. Обеспечить семантическую поддержку целостности Вашей базы данных. Для каждого поля базы данных задать следующие (подходящие по смыслу) виды декларативных ограничений целостности: - ограничения целостности атрибута: значение по умолчанию, задание обязательности или необязательности значений (Null), задание условий на значения атрибутов; - задание значения по умолчанию.

Лабораторная работа № 6.

Тема: Разработка информационной системы для работы с базой данных

Цель работы: Приобретение навыков доступа к базам данных в сети Интернет, используя возможности PHP. Задачами лабораторной работы являются овладение навыками создания и заполнения таблиц баз данных, создания представлений, триггеров и хранимых процедур, освоение программных технологий доступа к базам данных MySQL с помощью серверных сценариев PHP.

Теоретическая часть СУБД MySQL. Первоначально сервер MySQL разрабатывался для управления большими базами данных с целью обеспечить более высокую скорость работы по сравнению с существующими на тот момент аналогами. И вот уже в течение нескольких лет данный сервер успешно используется в условиях промышленной эксплуатации с высокими требованиями. Несмотря на то, что MySQL постоянно совершенствуется, он уже сегодня обеспечивает широкий спектр полезных функций. Благодаря своей доступности, скорости и безопасности MySQL очень хорошо подходит для доступа к базам данных по Интернету. GPL. MySQL - это программное обеспечение с открытым кодом. Это означает, что применять и модифицировать его может любой желающий. Такое ПО можно получать по Internet и использовать бесплатно. При этом каждый пользователь может изучить исходный код и изменить его в соответствии со своими потребностями. Использование программного обеспечения MySQL регламентируется лицензией GPL (GNU General Public License), в которой указано, что можно и чего нельзя делать с этим программным обеспечением в различных ситуациях. MySQL можно загрузить с веб-сайта http://www.mysql.com/. Учетные записи MySQL. У MySQL есть собственный интерфейс для организации взаимодействия с клиентами, с помощью которого можно перемещать данные и

изменять параметры баз данных. Чтобы иметь возможность работать с базой данных, необходимы учетная запись и пароль. Каждый сервер MySQL может содержать несколько баз данных, где группируются таблицы. Если MySQL установлен на локальном компьютере, то по умолчанию именем пользователя является root. Форматы таблиц MySQL. В MySQL можно было выбирать из семи основных форматов таблиц, наиболее распространенными из которых являются ISAM и InnoDB. Принятым по умолчанию типом таблиц в MySQL является MyISAM. Если попытаться воспользоваться таблицей, которая не была активизирована или добавлена при компиляции, MySQL вместо нее создаст таблицу типа MyISAM. Это очень полезная функция, когда необходимо произвести копирование таблиц с одного SQL- сервера на другой, а серверы поддерживают различные типы таблиц (например, при копировании таблиц на подчиненный компьютер, который оптимизирован для быстрой работы без использования транзакционных таблиц). Таблицы InnoDB в MySQL снабжены обработчиком таблиц, обеспечивающим безопасные транзакции (уровня ACID) с возможностями фиксации транзакции, отката и восстановления после сбоя. Для таблиц InnoDB осуществляется блокировка на уровне строки, а также используется метод чтения без блокировок в команде SELECT. Перечисленные функции позволяют улучшить взаимную совместимость и повысить производительность в многопользовательском режиме. В InnoDB нет необходимости в расширении блокировки, так как блоки строк в InnoDB занимают очень мало места. Важно, что для таблиц InnoDB поддерживаются ограничивающие условия FOREIGN KEY. Инструментарий phpMyAdmin. Инструмент phpMyAdmin позволяет администрировать MySQL с помощью обычного браузера. Все, что требуется для работы с этим инструментом, - это веб-сервер с установленным PHP и база данных MySQL, которую нужно администрировать. Инструмент администрирования позволяет увидеть параметры настройки базы данных и имеющиеся в ней объекты (например, таблицы), а также добавлять новые таблицы при помощи графического интерфейса. С помощью phpMyAdmin можно создавать новые базы данных и таблицы, запускать запросы и просматривать статистику работы сервера. Ограничения в MySQL. Важной особенностью (и недостатком) MySQL является отсутствие поддержки ограничений уровня столбцов таблиц (например, даже такое простое ограничение, как проверка принадлежности числа заданному диапазону). При этом реализация ограничений с помощью программного кода (запросов check) возможна, но не действенна: написанные запросы будут без проблем выполнены СУБД, но проигнорированы при работе с таблицами. Рассмотренные ранее СУБД Access и SQL Server решали задачу создания ограничений целостности с помощью соответствующих ограничений (типа Check), доступных как в графическом, так и программном (с помощью непосредственного ввода кода) режимах. Те же самые задачи в MySQL решаются намного сложнее и требуют написания соответствующих программных триггеров. Триггеры в MySQL. Поддержка для триггеров включена, начиная с MySQL 5.0.2. Триггер представляет собой именованный объект базы данных, который связан с таблицей, и он будет активизирован, когда специфическое событие происходит для таблицы. В общем виде программный код создания триггера имеет следующий вид: CREATE TRIGGER trigger_name trigger_time trigger_event ON tbl_name FOR EACH ROW trigger_stmt В приведенном фрагменте параметр trigger_time задает время действия. Это может быть BEFORE или AFTER, чтобы задать, что триггер активизируется прежде или после инструкции, которая активизировала это. Следующий параметр - trigger_event - указывает вид инструкции, которая активизирует триггер. Здесь trigger_event может быть одним из следующего: - INSERT: всякий раз, когда новая строка вставлена в таблицу. Например, через команды INSERT, LOAD DATA или REPLACE. - UPDATE: всякий раз, когда строка изменяется. Например, через инструкцию UPDATE. - DELETE: всякий раз, когда строка удалена из таблицы. Например, через инструкции DELETE и REPLACE. Важно, что не может быть двух триггеров для данной таблицы, которые имеют те же самые время действия и событие. Например, не может быть два триггера BEFORE UPDATE для таблицы, но возможны BEFORE UPDATE и BEFORE INSERT или BEFORE UPDATE и AFTER UPDATE. Следующий параметр - trigger_stmt -

задает инструкцию, которая будет выполнена, когда триггер активизируется. Если нужно выполнить много инструкций, используется операторная конструкция BEGIN ... END. Это также дает возможность использовать те же самые инструкции, которые являются допустимыми внутри сохраненных подпрограмм. Схема взаимодействия серверного php-приложения с базой данных MySQL по приведенному алгоритму проиллюстрирована на рис. 1.

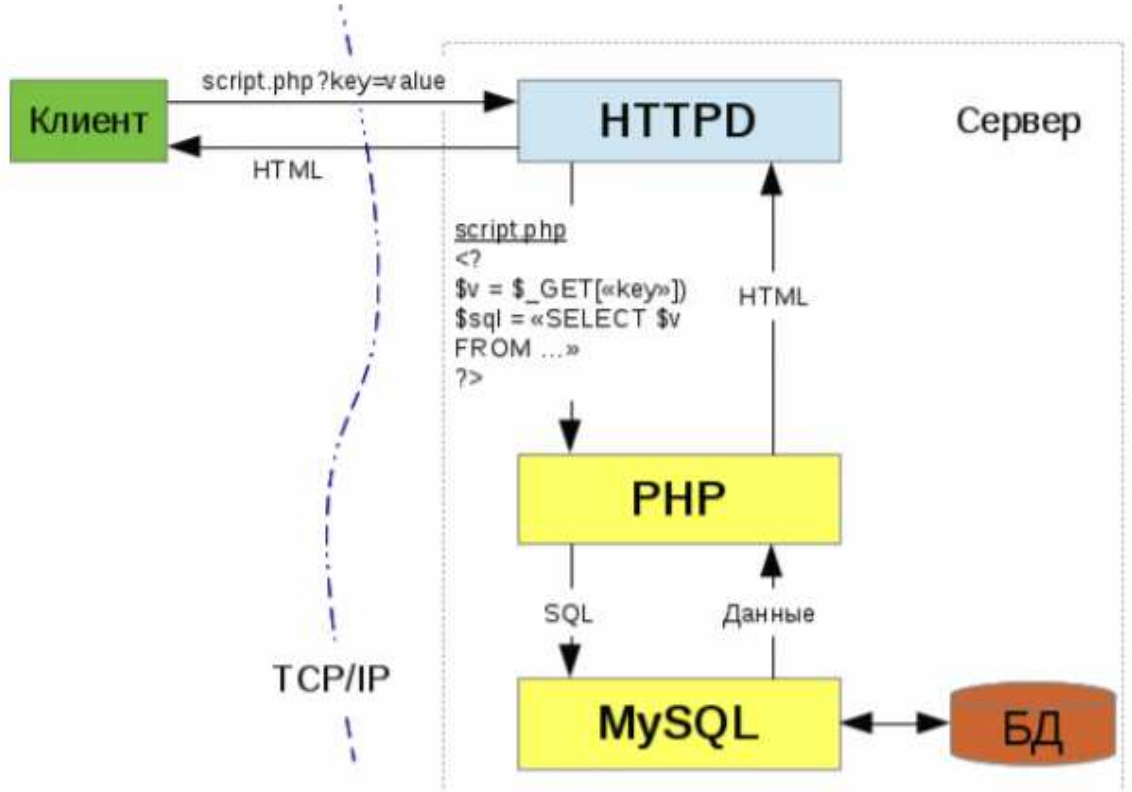


Рис. 24. Взаимодействие php-скрипта и БД MySQL

MySQL и PHP. Язык PHP (PHP: Hypertext Preprocessor) представляет собой язык сценариев, которые внедряются в страницы HTML для исполнения на стороне сервера. Как правило, для отделения PHP-кода от кода HTML используются символы , реже - и , а также инструкции «в стиле ASP» - . Web-страница PHP имеет расширение .php и состоит из трех разделов:

- директивы страницы - используются для настройки и определяют, как должна обрабатываться страница. Например, так можно задать подключение внешних файлов.

- код - программный код, реализующий выполняемые на сервереоперации.

- разметка страницы - это HTML-код страницы, включающий тег body и его содержимое. Поддержка PHP обеспечивается многими серверами, но традиционно такие Web-приложения функционируют на Web-сервере Apache. Приложения PHP, использующие для хранения информации базу данных, как правило, работают с системой управления базами данных MySQL. Для работы с базами данных PHP располагает достаточно широким набором функций (табл. 2) - от установки соединения с базой данных до извлечения отдельных значений, полученных в результате выполнения запроса.

Таблица 2 - Функции взаимодействия с базами данных в PHP-приложениях (СУБД MySQL)

Название класса	Описание класса
mysql_connect	Соединение с источником данных
mysql_select_db	Выбор базы данных для дальнейшей работы
mysql_query	Отправка SQL-запросов серверу
mysql_fetch_array	Представление результата запроса в виде ассоциативного массива
mysql_result	Доступ к отдельному полю записи результата запроса

Задание.

1. Запустить СУБД MySQL:

Рабочий стол | ярлык Start Denwer ►окно запуска Denwer; дождаться закрытия окна

2. Открыть среду PHPMyAdmin:

Запустить браузер ►адресная строка ← http://localhost ►окн. «Ура, заработало!» | выбр.

http://localhost/Tools/phpMyAdmin ► окн. phpMyAdmin; Server: localhost

3. Создать новую базу данных Education:

ст. MySQL localhost | Create database: | ввести Education | Collation ← utf8_unicode_ci | кн. Create

►окн. Server: localhost; Database: Education |

Database Education has been created.

4. В базе данных с помощью конструктора таблиц создать структуру таблицы Students (справочник

студентов):

- создать новую таблицу Students с 11 полями:

окн. Server: localhost; Database: Education | Create new table on database Education: | Name ←

Students; Number of fields ← 11 | кн. Go ► окн. Server: localhost; Database: Education; Table:

Students;

- задать поле идентификатора студента (счетчик, первичный ключ):

ст. Field ← stud_ID; □□□□← int; уст. A_I; Index ← □rim□r□; перейти на след. строку;

- задать поле номера зачетной книжки (уникальное, 6 цифр): ст. Field ← n□_zk; □□□□ ← v□r□h□r;

Length / Values1 ← 6; Ind□□← Uniqu□; перейти на след. строку;

- задать поля фамилии, имени, отчества (строковые):

ст. Field ← f□m; □□□□ ← v□r□h□r; L□ngth / V□lu□s1 ← 20; перейти на след. строку; аналогично - для

имени (im) и отчества (ot);

- создать уникальный индекс ФИО из полей fam, im, ot:

ст. Field fam ← Ind□□← Ind□□; перейти на след. строку; ст. Field im ← Ind□□← Ind□□; перейти на

след. строку; ст. Field ot ← Ind□□← Ind□□;

- задать поле кода специальности (до 7 букв):

ст. Field ← s□□□; □□□□← v□r□h□r; L□ngth / V□lu□s1 ← 7; перейти на след. строку;

- задать поле курса (одна цифра от 1 до 6):

ст. Field ← kurs; □□□□← v□r□h□r; L□ngth / V□lu□s1 ← 1; перейти на след. строку;

- задать поле номера группы (две цифры):

ст. Field ← gr; □□□□← v□r□h□r; L□ngth / V□lu□s1 ← 2; перейти на след. строку;

- задать поле даты рождения:

ст. Field ← data_r; Type ← d□t□; перейти на след. строку;

- задать поле биографии (мемо):

ст. Field ← bi□gr; □□□□← l□ngt□□t; уст. Null; перейти на след. строку;

- задать поле фотографии (объект):

ст. Field ← f□t□; □□□□← bl□b; уст. Null; перейти на след. строку;

кн. Save ► окн. Table `Education`.`Students` has been created.

3. Аналогичным образом создать структуру таблицы Subjects (справочник предметов):

- создать новую таблицу базы данных Education: пан. phpMyAdmin | Database ← Education (1);

пан. Create new table on database Education: | Name ← Subj□□ts; Numb□r □f fi□lds ← 5 | кн. Go ►

окн. Server: localhost; Database: Education; Table: Subjects.

- задать поле идентификатора предмета (счетчик, первичный ключ):

ст. Field ← □r□dm_ID; □□□□← int; уст. A_I; Index ← □rim□r□; перейти на след. строку;

- аналогичным образом создать остальные поля таблицы:

- name (название предмета (строковое));

- cycle (цикл дисциплин, к которым относится предмет (строковое));

- hours (количество часов (числовое));

- dep (название кафедры, на которой ведется преподавание предмета (строковое));

- сохранить структуру таблицы:

кн. Save ► окн. Table `Education`.`Subjects` has been created.

4. Аналогичным образом создать структуру таблицы Uspev (успеваемость студентов по предметам)

с полями:

- stud (идентификатор студента (тип такой же, как в таблице Студенты, но не счетчик));

- predm (идентификатор предмета (тип такой же, как в таблице Предметы, но не счетчик));

- ocenka (оценка (символ));

- data (дата (дата)).

- ввести составной первичный ключ (поля идентификаторов студента ипредмета): Поле stud | Index

← □rim□r□; перейти на след. строку;

Поле predm | Index ← Primary;

- сохранить структуру таблицы:

кн. Save ► окн. Table `Education`.`Uspev` has been created.

5. Задать ограничения для столбцов таблицы Students в форме триггера students_constraints,

срабатывающего при попытке вставки новой записи в таблицу:

- создать новый SQL-запрос для таблицы Students:

пан. phpMyAdmin | Database: ← □du□□ti□n (1); выбр. Students; вкл. SQL ►

окн. Run SQL query/queries on database Education | удалить текст «SELECT * FROM `Students`

WHERE 1»;

- ввести SQL-запрос на удаление триггера students_constraints_insert в случае его существования

(для возможности редактирования текста триггера):

DROP TRIGGER IF EXISTS `students_constraints_insert` ;

- ввести «заголовочный» код создания триггера: delimiter ||

CREATE TRIGGER `students_constraints_insert` BEFORE INSERT ON `Students` FOR EACH

ROW BEGIN

- ввести код, проверяющий, что значение поля kurs лежит в пределах от 1 до 6 и заменяющий его на

0в противном случае:

IF not(NEW.kurs >=1 and NEW.kurs <= 6) THEN SET NEW.kurs = 0;

END IF;

- ввести код, проверяющий, что значение поля spec вводится русскими буквами и заменяющийего

на 0 в противном случае:

IF not(NEW.spec >= 'A' AND NEW.spec <= 'я') THEN SET NEW.spec = 0;

END IF;

- ввести код, проверяющий, что значение поля gr лежит в пределах от 1 до 99 и заменяющий его на

0 в противном случае:

IF not(NEW.gr >='01' AND NEW.gr <= '99') THEN SET NEW.gr = 0;

END IF;

- ввести завершающий код триггера:

END || delimiter ;

6. Аналогичным образом запрограммировать триггер students_constraints_update, срабатывающий

при попытке изменения записи в таблице.

7. Для таблиц Subjects и Uspev написать по два триггера, проверяющие целостность данных при

вставке и изменении данных.

8. Установить связи между таблицами и указать правила ссылочной целостности:

- изменить тип таблицы Student:

пан. phpMyAdmin | Database: ← □du□□ti□n (1); выбр. Students; вкл. Operations ► Storage Engine

← Inn□D□;

аналогично для всех остальных таблиц.

- указать поле stud в качестве внешнего ключа таблицы Students, ссылающегося на первичный ключ

таблицы Students:

пан. phpMyAdmin | Database: ← □du□□ti□n (3); выбр. Uspev; вкл. Operations ►

выбр. Relation View ► ф. Links to | стр. stud; Foreign key (InnoDB) ← `Education`.`Students`;

onDelete ← □s□d□; □nU□d□t□← □s□d□;

аналогичным образом задать внешний ключ, ссылающийся на первичный ключ таблицы Subjects.

8. Наполнить базу данных сведениями о студентах (не менее 5), предметах (не менее 3) и оценках (не

менее 10):

пан. phpMyAdmin | Database: ← □du□□ti□n (3); выбр. таблицу; вкл. Insert ►заполнить столбец

Value.

Освоить приемы изменения и удаления полей и записей. Проверить работоспособность ограничений

значений полей, уникальности и др., предусмотренные при задании структуры базы данных. Проверить

работоспособность ссылочной целостности, удаляя, изменяя и вставляя данные.

Б. Представления и хранимые процедуры

1. Создать просмотр для вывода кратких сведений о студентах (идентификатор, номер зачетки,

фамилия и инициалы, идентификатор группы):

- создать новый SQL-запрос для таблицы Students:

пан. phpMyAdmin | Database: ← □du□□ti□n (3); выбр. Students; вкл. SQL ►

окн. Run SQL query/queries on database Education | удалить текст «SELECT * FROM `Students`

WHERE 1»;

- ввести программный код запроса на извлечение данных:

SELECT `stud_ID`,`no_zk`,CONCAT(fam,' ',SUBSTRING(im,1),'.',SUBSTRING(ot,1),'.') AS FIO,

CONCAT(spec,'-',kurs,gr) AS Gruppa FROM `Students` WHERE 1

- выполнить запрос

окн. Run SQL query/queries on database Education | кн. Go ► результаты на экране;

- сформировать представление по результатам запроса:

ф. Query results operations | CREATE VIEW | VIEW name ← Students_info | кн. Go.

2. Аналогичным образом создать представление для вывода сведений об успеваемости студентов из

таблицы успеваемости с указанием сведений о студенте из запроса Students_info и сведений о предмете

из таблицы предметов. Результат должен содержать следующие поля: ФИО студента, Группа, Предмет,

Дата, Оценка. Назвать представление как Students_uspev. Программный код запроса:

SELECT `no_zk`,`FIO`,`Gruppa`,`ocenka`,`data` FROM `students_info`,`uspev`,`subjects` WHERE

`students_info`.`stud_ID`=`uspev`.`stud ` AND

`subjects`.`predm_ID`=`uspev`.`predm `

3. Создать хранимую процедуру для вывода кратких сведений о студенте (идентификатор, фамилия

и инициалы, идентификатор группы) по номеру его зачетной книжки:

- создать новый SQL-запрос для базы данных Education:

пан. phpMyAdmin | Database: ← □du□□ti□n (3); вкл. SQL ► окн. Run SQL query/queries on database

Education;

- ввести программный код хранимой процедуры: DELIMITER $$

CREATE PROCEDURE studs(IN IN_ZK VARCHAR(6),OUT V_ID INT, OUT V_ZK

VARCHAR(6),OUT V_FIO VARCHAR(25),OUT V_GR VARCHAR(25))

BEGIN

SELECT stud_id, no_zk, concat(fam,' ',substring(im,1),'.',substring(ot,1),'.'), concat(spec,'-',kurs,gr)

into V_ID, V_ZK, V_FIO, V_GR FROM Students

WHERE `no_zk` = IN_ZK; END;

$$ DELIMITER ;

- проверить работоспособность хранимой процедуры:

пан. phpMyAdmin | Database: ← □du□□ti□n (3); вкл. SQL ► окн. Run SQL query/queries on database

Education; ввести код:

CALL studs('номер зачетной книжки',@V_ID,@V_ZK,@V_FIO,@V_GR); SELECT

@V_ID,@V_ZK,@V_FIO,@V_GR;

кн. Go ► Результат на экране.

B. PHP и MySQL

1. Запустить программную оболочку Denwer:

Рабочий стол | ярлык Start Denwer ►окно запуска Denwer; дождаться закрытия окна

2. Проверить корректность работы Denwer:

Запустить браузер ►адресная строка ← http://localhost ►окн. «Ура, заработало!»

3. Создать виртуальную директорию для хранения файлов веб-приложения: Мой компьютер |

Локальный диск D | WebServers | denwer | www | denwer | создать папку под именем lab3_группа

(например, lab3_asoi338).

При наличии уже существующей папки с таким именем добавить постфикс «2», например,

lab3_asoi338_2.

4. Создать файл index.php начальной страницы веб-приложения: Директория lab3_группа | Создать

текстовый документ | Переименовать файл как index.php

5. Открыть файл index.php с помощью редактора Notepad++: Директория lab3_группа | Файл

index.php | пр. кн. мыши; выбр. Edit with Notepad++ ►окно редактора Notepad++

6. Ввести HTML-разметку страницы index.php:

```
<HTML>
<HEAD><TITLE>Лабораторная работа 3</TITLE></HEAD>
<BODY>
<H2>Информация о студентах</H2>
<FORM id="form" method="POST" action="index.php">
<TABLE border="1" width="60%">
<TR>
<TH width="10%">Код</TH>
<TH width="20%">Зачетная книжка</TH>
<TH width="40%">ФИО</TH>
<TH width="30%">Группа</TH>
</TR>
<TR align="center">
<TD width="10%">Значение кода</TD>
<TD width="20%">Значение зачетки</TD>
<TD width="40%">Значение ФИО</TD>
<TD width="30%">Значение группы</TD>
</TR>
</TABLE>
<BR/> Номер зачетной книжки: <input name="zk" type="text"/>
<input type="submit" value="Запрос"/>
</FORM>
</BODY>
</HTML>
```

7. Проверить работоспособность созданной страницы:

Браузер со стартовой страницей Denwer | строка адреса «http://localhost/denwer/» | дополнить

названием виртуальной директории (см. п. 3),

например, как http://localhost/denwer/ lab3_asoi338 ►страница на экране

8. Создать PHP-сценарий соединения с базой данных Education:

- создать в виртуальной директории и открыть файл connection.php: Директория lab3_группа |

Создать текстовый документ | Переименовать файл как connection.php | пр. кн. мыши; выбр. Edit with Notepad++ ►окно редактора Notepad++

- ввести программный код сценария:

- ввести открывающий тег сценария:

```
<?php
```

- ввести программный код соединения с локальным сервером:

```
$link = @mysql_connect("localhost", "root")
or die("Невозможно соединиться с сервером");
```

- ввести программный код соединения с базой данных Education:

```
$db=@mysql_select_db("Education") or die("Нет такой базы данных");
```

- ввести закрывающий тег сценария:

```
?>
```

- сохранить сценарий.

9. Дополнить файл index.php PHP-инструкциями:

- подключить сценарий соединения с базой данных:

окно редактора Notepad++ | вкл. Index.php | поставить курсор до первой строки <HTML> | ввести

код:

```
<?php include("connection.php");
?>
```

- проверить работоспособность сценария, обновив в браузере страницу index.php.

- выполнить запрос к представлению Students_info базы данных: окно редактора Notepad++ | вкл.

Index.php | поставить курсор после строки include("connection.php"); | ввести код:

```
$sql = "SELECT * FROM `students_info`";
$query = mysql_query($sql);
```

- организовать цикл по строкам таблицы:

окно редактора Notepad++ | вкл. Index.php | поставить курсор перед второй строкой <TR> |

ввести код:

```
<?php
for($i=0;$i<$count;$i++)
{
?>
```

поставить курсор после второй строки </TR> | ввести код:

```
<?php
}
?>
```

- выполнить подстановку результатов запроса в строки таблицы: окно редактора Notepad++ | вкл.

Index.php | поставить курсор

перед второй строкой с текстом «Значение кода» | заменить текст «Значение кода» на:

```
<?php echo mysql_result($query,$i,stud_id);?>
```

аналогичным образом заменить фрагменты текста «Значение зачетки»,

«Значение ФИО» и «Значение группы» на фрагменты кода:

```
<?php echo mysql_result($query,$i,no_zk);?>
<?php echo mysql_result($query,$i,FIO);?>
<?php echo mysql_result($query,$i,Gruppa);?>
```

- сохранить файл index.php.

- проверить работоспособность сценария, обновив в браузере страницу index.php.

- добавить инструкции фильтрации данных по номеру зачетной книжки:

- добавить условие, проверяющее, был ли запрос на фильтрацию: окно редактора Notepad++ |

вкл. Index.php | поставить курсор

перед строкой с текстом «sql = "SELECT * FROM `students_info`";» | ввести код:

```
if(!($_POST['zk']) or $_POST['zk']=='')
{
```

- закрыть условный блок (если запроса не было, то будут показаны все записи): окно редактора

Notepad++ | вкл. Index.php | поставить курсор

после строки с текстом «$count = mysql_num_rows($query);» | ввести «}»

- добавить код, выполняющий запрос на фильтрацию:

окно редактора Notepad++ | вкл. Index.php | поставить курсор после строки с текстом «};»

| ввести код

```
else
{
$sql = "SELECT * FROM `students_info` where
`no_zk`='".$_POST['zk']."'";
$query = mysql_query($sql);
$count = mysql_num_rows($query);
}
```

- сохранить файл index.php.
- проверить работоспособность сценария, обновив в браузере страницу index.php.

Лабораторная работа № 7.

Тема: «Создание SQL-запросов»

Цель работы: Научиться создавать запросы.

Формируемые компетенции или их части

Теоретическая часть

Запрос SQL — это запрос, создаваемый при помощи инструкций SQL. Язык SQL (Structured Query Language) используется при создании запросов, а также для обновления и управления реляционными БД. В среде MS Access, когда пользователь создает запрос в режиме конструктора запроса (с помощью построителя запросов), MS Access автоматически создает эквивалентную инструкцию SQL. Фактически, для большинства свойств запроса, доступных в окне свойств в режиме конструктора, имеются эквивалентные предложения или параметры языка SQL, доступные в режиме SQL. При необходимости, пользователь имеет возможность просматривать и редактировать инструкции SQL в режиме SQL. После внесения изменений в запрос в режиме SQL его вид в режиме конструктора может измениться. Некоторые запросы, которые называют запросами SQL, невозможно создать в бланке запроса. Для запросов к серверу, управляющих запросов и запросов на объединение необходимо создавать инструкции SQL непосредственно в окно запроса в режиме SQL. Для подчиненного запроса пользователь должен ввести инструкцию SQL в строку Поле или Условие отбора в бланке запроса. Синтаксиса написания SQL-предложений: в описании команд слова, написанные прописными латинскими буквами, являются зарезервированными словами SQL; фрагменты SQL-предложений, заключенные в фигурные скобки и разделенные символом « », являются альтернативными. При формировании соответствующей команды для конкретного случая необходимо выбрать одну из них;
фрагмент описываемого SQL-предложения, заключенный в квадратные скобки [ ], имеет необязательный характер и может не использоваться; многоточие …, стоящее перед закрывающейся скобкой, говорит о том, что фрагмент, указанный в этих скобках, может быть повторен;

2 Описание команд SQL Выборка записей Инструкция SELECT. При выполнении инструкции SELECT СУБД находит указанную таблицу или таблицы, извлекает заданные столбцы, выделяет строки, соответствующие условию отбора, и сортирует или группирует результирующие строки в указанном порядке в виде набора записей. Синтаксис команды:

SELECT [предикат] { * | таблица.* | [таблица.]поле_1 [AS псевдоним_2] [, [таблица.]поле_2[AS

псевдоним_2] [, ...]]} FROM выражение [, ...] [WHERE...] [GROUP BY...] [HAVING...] [ORDER BY...]

где предикат — один из следующих предикатов отбора: ALL, DISTINCT, DISTINCTROW, TOP.

Данные ключевые слова используются для ограничения числа возвращаемых записей. Если они отсутствуют, по умолчанию используется предикат ALL; * указывает, что результирующий набор записей будет содержать все поля заданной таблицы или таблиц. Следующая инструкция отбирает все

поля из таблицы «Студенты»:

SELECT * FROM Студенты;

таблица — имя таблицы, из которой выбираются записи; поле_1, поле_2 — имена полей, из которых должны быть отобраны данные; псевдоним_1, псевдоним_2 — ассоциации, которые станут заголовками столбцов вместо исходных названий полей в таблице; выражение — имена одной или нескольких таблиц, которые содержат необходимые для отбора записи; предложение GROUP BY в SQLпредложении объединяет записи с одинаковыми значениями в указанном списке полей в одну запись.

Если инструкция SELECT содержит статистическую функцию SQL, например Sum или Count, то для каждой записи будет вычислено итоговое значение; предложение HAVING определяет, какие сгруппированные записи, выданные в результате выполнения запроса, отображаются при использовании инструкции SELECT с предложением GROUP BY. После того как записи результирующего набора будут сгруппированы с 3 помощью предложения GROUP BY, предложение HAVING отберет те из них, которые удовлетворяют условиям отбора, указанным в предложении HAVING; предложение ORDER BY позволяет отсортировать записи, полученные в результате запроса, в порядке возрастания или убывания на основе значений указанного поля или полей. Следует отметить, что инструкции SELECT не изменяют данные в базе данных. Приведем минимальный синтаксис инструкции SELECT: SELECT поля FROM таблица. Если несколько таблиц, включенных в предложение

FROM, содержат одноименные поля, перед именем такого поля следует ввести имя таблицы и оператор « . » (точка). Предположим, что поле «Номер_группы» содержится в таблицах «Студенты» и «Группы». Следующая инструкция SQL отберет поле «Номер_группы» и «ФИО_студента» из таблицы «Студенты» и «ФИО_куратора» из таблицы «Группы» при номере группы, равном 432-1: SELECT Группы.Номер_группы, Группы.ФИО_куратора, Студенты.ФИО_студента FROM

Группы, Студенты WHERE Группы.Номер_группы = Студенты.Номер_группы AND

На рис. 1 приведен пример выполнения данного запроса. Таблицы БД СТУДЕНТЫ

Номер_зачетной книжки	ФИО_студента	Дата рождения	Место рождения	Номер_группы
1992412-11	Карасев А.А.	27.08.75	г. Чита	412-1
1992432-11	Данилов О. В.	27.08.75	г. Алматы	432-1
1992432-12	Раевский А. И.	20.05.75	г. Бишкек	432-1
1992432-22	Глазов О.А	04.07.75	г. Киров	432-1

ГРУППЫ

Номер_Группы	ФИО_куратора
412-1	Самойлов С.С.
432-1	Авдеев Р.М

Результат выполнения запроса

Номер_группы	ФИО_куратора	ФИО_студента
432-1	Авдеев Р.М	Данилов О. В.
432-1	Авдеев Р.М	Раевский А. И.
432-1	Авдеев Р.М	Глазов О.А

Рис. 1. Пример выполнения запроса на выборку

Помимо обычных знаков сравнения (=,,<=,>=, <>) в языке SQL в условии отбора используются

ряд ключевых слов:

Is not null — выбрать только непустые значения;

Is null — выбрать только пустые значения;

Between … And определяет принадлежность значения выражения указанному диапазону.

Синтаксис:

выражение [Not] Between значение_1 And значение_2 ,

где выражение — выражение, определяющее поле, значение которого проверяется на

принадлежность к диапазону;

значение_1, значение_2 – выражения, задающие границы диапазона.

Если значение поля, определенного в аргументе выражение, попадает в диапазон, задаваемый

аргументами значение_1 и значение_2 (включительно), то оператор Between...And возвращает значение

True; в противном случае возвращается значение False.

Логический оператор Not позволяет проверить противоположное условие: что выражение

находится за пределами диапазона, заданного с помощью аргументов значение_1 и значение_2.

Оператор Between...And часто используют для проверки: попадает ли значение поля в указанный

диапазон чисел.

В следующем примере выдается список студентов, получающих стипендию от 800 до 900

рублей:

SELECT ФИО_студента, Размер_стипендии

FROM Студенты

WHERE Размер_стипендии Between 800 And 900

На рис. 2 приведен результат выполнения запроса. СТУДЕНТЫ

Номер_зачетной книжки	ФИО_студента	Размер_стипендии
1992412-11	Карасев А.А.	900
1992432-11	Данилов О. В.	800
1992432-12	Раевский А. И.	950
1992432-22	Глазов О.А	850

Результирующий набор данных

ФИО_студента	Размер_стипендии
Карасев А.А.	900
Данилов О. В.	800
Глазов О.А	850

Если выражение, значение_1 или значение_2 имеет значение Null, оператор Between...And возвращает значение Null. Оператор Like используется для сравнения строкового выражения с образцом. Синтаксис: выражение Like «образец», где выражение — выражение SQL, используемое в предложении WHERE; образец — строка, с которой сравнивается выражение. Оператор Like используется для нахождения в поле значений, соответствующих указанному образцу. Для аргумента образец можно задавать полное значение (например, Like «Иванов») или использовать подстановочные знаки для поиска диапазона значений (например, Like "Ив*"). Приведем перечень подстановочных символов и пример их использования в языке Jet SQL согласно документации Microsoft. Параметры оператора Like

Тип совпадения	Образец	Совпадение (True)	Несовпаде-ние (False)
Несколько символов	a*a	aa, aBa, aBBBa	aBC
	ab	abc, AABB, Xab	aZb, bac
Специальный символ	a[*]a	a*a	aaa
Несколько символов	ab*	abcdefg, abc	cab, aab
Одиночный символ	a?a	aaa, a3a, aBa	aBBBa
Одиночная цифра	a#a	a0a, a1a, a2a	aaa, a10a
Диапазон символов	[a-z]	f, p, j	2, &
Вне диапазона	[!a-z]	9, &, %	b, a
Не цифра	[!0-9]	A, a, &, ~	0, 1, 9
Комбинированное выражение	a[!b-m]#	An9, az0, a99	abc, aj0

Внутреннее соединение Операция INNER JOIN объединяет записи из двух таблиц, если связующие поля этих таблиц содержат одинаковые значения. Синтаксис операции: FROM таблица_1 INNER JOIN таблица_2 ON таблица_1.поле_1 оператор таблица_2.поле_2 где таблица_1, таблица_2 — имена таблиц, записи которых подлежат объединению; поле_1, поле_2 — имена объединяемых полей. Поля должны иметь одинаковый тип данных и содержать данные одного рода, однако эти поля могут иметь разные имена; оператор — любой оператор сравнения: "=," " 4) Предикат ALL используется для отбора в главном запросе только тех записей, которые удовлетворяют сравнению со всеми записями, отобранными в подчиненном запросе. Если в предыдущем примере предикат ANY заменить предикатом ALL, результат запроса будет включать только тех студентов, у которых средний балл больше 4. Это условие является значительно более жестким. Предикат IN используется для отбора в главном запросе только тех записей, которые содержат значения, совпадающие с одним из отобранных подчиненным запросом. Следующий пример возвращает сведения обо всех студентах, средний балл которых за семестр был больше 4. SELECT * FROM Студенты WHERE Номер_зачетной_книжки in (SELECT Номер_зачетной_книжки FROM Успеваемость WHERE оценка > 4) Предикат NOT IN используется для отбора в главном запросе только тех записей, которые содержат значения, не совпадающие ни с одним из отобранных подчиненным запросом. Предикат EXISTS (с необязательным зарезервированным словом NOT) используется в логическом выражении для определения того, должен ли подчиненный запрос возвращать какие-либо записи. В подчиненном запросе можно использовать псевдонимы таблиц для ссылки на таблицы, перечисленные в предложении FROM, расположенном вне подчиненного запроса. В следующем примере отбираются фамилии и имена студентов, чья стипендия равна или больше средней стипендии студентов, обучающихся в той же группе. В данном примере таблица СТУДЕНТЫ получает псевдоним С1: SELECT Фамилия, Имя, Номер_группы, Стипендия FROM СТУДЕНТЫ AS С1 WHERE Стипендия >= (SELECT Avg(Стипендия) FROM СТУДЕНТЫ WHERE С1.Номер_группы = СТУДЕНТЫ.Номер_группы) Order by Номер_группы;

В последнем примере зарезервированное слово AS не является обязательным. Некоторые

подчиненные запросы можно использовать в перекрестных запросах как предикаты (в предложении

WHERE). Подчиненные запросы, используемые для вывода результатов (в списке SELECT), нельзя

использовать в перекрестных запросах. Создание новой таблицы Инструкция CREATE TABLE создает

новую таблицу и используется для описания ее полей и индексов. Если для поля добавлено ограничение

NOT NULL, то при добавлении новых записей это поле должно содержать допустимые данные.

Синтаксис:

CREATE TABLE таблица (поле_1 тип [(размер)] [NOT NULL] [индекс_1] [, поле_2 тип [(размер)]

[NOT NULL] [индекс_2] [, ...]] [, CONSTRAINT составной Индекс [, ...]]),

где таблица — имя создаваемой таблицы; поле_1, поле_2 — имена одного или нескольких полей, создаваемых в новой таблице.

Таблица должна содержать хотя бы одно поле; тип — тип данных поля в новой таблице; размер — размер поля в символах (только для текстовых и двоичных полей);

индекс_1, индекс_2 — предложение CONSTRAINT, предназначенное для создания простого индекса; составной Индекс — предложение CONSTRAINT, предназначенное для создания составного индекса. следующем примере создается новая таблица с двумя полями:

CREATE TABLE Студенты (Номер_зачетной_книжки integer PRIMARY KEY, ФИО_студента TEXT (50), Место_рождения TEXT (50));

В результате выполнения этого запроса будет создана таблица со следующей схемой (рис. 5):

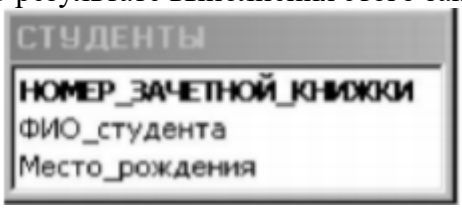


Рис. 5. Схема таблицы СТУДЕНТЫ Предложение CONSTRAINT используется в инструкциях ALTER TABLE и CREATE TABLE для создания или удаления индексов. Существуют два типа предложений CONSTRAINT: для создания простого индекса (по одному полю) и для создания составного индекса (по нескольким полям). Синтаксис: простой индекс: CONSTRAINT имя {PRIMARY KEY|UNIQUE | NOT NULL]} составной индекс: CONSTRAINT имя {PRIMARY KEY (ключевое_1[, ключевое_2 [, ...]]) | UNIQUE (уникальное_1[, уникальное_2 [, ...]]) | NOT NULL (непустое_1[, непустое_2 [, ...]]) | FOREIGN KEY (ссылка_1[, ссылка_2 [, ...]]) REFERENCES внешняя Таблица [(внешнее Поле_1 [, внешнее Поле_2 [, ...]])]}, где имя — имя индекса, который следует создать; ключевое_1, ключевое_2 — имена одного или нескольких полей, которые следует назначить ключевыми; уникальное_1, уникальное_2 — имена одного или нескольких полей, которые следует включить в уникальный индекс; непустое_1, непустое_2 — имена одного или нескольких полей, в которых запрещаются значения Null; ссылка_1, ссылка_2 — имена одного или нескольких полей, включенных во внешний ключ, которые содержат ссылки на поля в другой таблице; внешняя Таблица — имя внешней таблицы, которая содержит поля, указанные с помощью аргумента внешнееПоле; внешнее Поле_1, внешнее Поле_2 — имена одного или нескольких полей во внешней Таблице, на которые ссылаются поля, указанные с помощью аргумента ссылка_1, ссылка_2. Это предложение можно опустить, если данное поле является ключом внешней Таблицы. Предложение CONSTRAINT позволяет создать для поля индекс одного из двух описанных ниже типов: 1) уникальный индекс, использующий для создания зарезервированное слово UNIQUE. Это означает, что в таблице не может быть двух записей, имеющих одно и то же значение в этом поле. Уникальный индекс создается для любого поля или любой группы полей. Если в таблице определен составной уникальный индекс, то комбинация значений включенных в него полей должна быть уникальной для каждой записи таблицы, хотя отдельные поля и могут иметь совпадающие значения; 2) ключ таблицы, состоящий из одного или нескольких полей, использующий зарезервированные слова PRIMARY KEY. Все значения ключа таблицы должны быть уникальными и не значениями Null. Кроме того, в таблице может быть только один ключ. Для создания внешнего ключа можно использовать зарезервированную конструкцию FOREIGN KEY. Если ключ внешней таблицы состоит из нескольких полей, необходимо использовать предложение CONSTRAINT. При этом следует перечислить все поля, содержащие ссылки на

поля во внешней таблице, а также указать имя внешней таблицы и имена полей внешней таблицы, на которые ссылаются поля, перечисленные выше, причем в том же порядке. Однако, если последние поля являются ключом внешней таблицы, то указывать их необязательно, поскольку ядро базы данных считает, что в качестве этих полей следует использовать поля, составляющие ключ внешней таблицы. В следующем примере создается таблица ЗАДОЛЖЕННОСТЬ_ЗА_ОБУЧЕНИЕ с единственным полем НОМЕР_ЗАЧЕТНОЙ_КНИЖКИ и внешним ключом f1_i, связанным с полем НОМЕР_ЗАЧЕТНОЙ_КНИЖКИ в таблице СТУДЕНТЫ: CREATE TABLE Задолженность_за_обучение (Код_задолженности integer PRIMARY KEY, Номер_зачетной_книжки integer, CONSTRAINT f1_i FOREIGN KEY (Номер_зачетной_книжки) REFERENCES Студенты (Номер_зачетной_книжки)); Внешний вид схемы БД, состоящей из таблиц СТУДЕНТЫ и ЗАДОЛЖЕННОСТЬ_ЗА_ОБУЧЕНИЕ, представлен на рис. 6.

Рис. 6 Схема данных

Изменение структуры таблицы Инструкция ALTER TABLE изменяет структуру таблицы, созданной с помощью инструкции CREATE TABLE. Синтаксис: ALTER TABLE таблица {ADD {COLUMN поле тип[(размер)] [NOT NULL] [CONSTRAINT индекс] | CONSTRAINT составной Индекс} | DROP {COLUMN поле I CONSTRAINT имя Индекса} } где таблица — имя изменяемой таблицы; поле— имя поля, добавляемого в таблицу или удаляемого из нее; тип — тип данных поля; размер — размер поля; индекс — индекс для поля; составной Индекс — описание составного индекса, добавляемого к таблице; имя Индекса — имя составного индекса, который следует удалить. С помощью инструкции ALTER TABLE существующую таблицу можно изменить несколькими способами: 1) добавить новое поле в таблицу с помощью предложения ADD COLUMN. В этом случае необходимо указать имя поля, его тип и размер. Например, следующая инструкция добавляет в таблицу СТУДЕНТЫ текстовое поле ПРИМЕЧАНИЯ длиной 50 символов: ALTER TABLE Студенты ADD COLUMN Примечания TEXT(50) Если для поля добавлено ограничение NOT NULL, то при добавлении новых записей это поле должно содержать допустимые данные; 2) добавить составной индекс с помощью зарезервированных слов ADD CONSTRAINT; 3) удалить поле с помощью зарезервированных слов DROP COLUMN. В этом случае необходимо указать только имя поля; 4) удалить составной индекс с помощью зарезервированных слов DROP CONSTRAINT. В этом случае указывается только имя составного индекса, следующее за зарезервированным словом CONSTRAINT. Создание индекса с помощью инструкции CREATE INDEX CREATE INDEX создает новый индекс для существующей таблицы. Синтаксис команды: CREATE [UNIQUE] INDEX индекс ON таблица (поле [ASC|DESC][, поле [ASC|DESC], ...]) [WITH { PRIMARY | DISALLOW NULL | IGNORE NULL }] где индекс — имя создаваемого индекса; таблица — имя существующей таблицы, для которой создается индекс; поле — имена одного или нескольких полей, включаемых в индекс. Для создания простого индекса, состоящего из одного поля, вводится имя поля в круглых скобках сразу после имени таблицы. Для создания составного индекса, состоящего из нескольких полей, перечисляются имена всех этих полей. Для расположения элементов индекса в убывающем порядке используется зарезервированное слово DESC; в противном случае будет принят порядок по возрастанию. Чтобы запретить совпадение значений индексированных полей в разных записях, используется зарезервированное слово UNIQUE. Необязательное предложение WITH позволяет задать условия на значения. Например: с помощью параметра DISALLOW NULL можно запретить значения Null в индексированных полях новых записей; параметр IGNORE NULL позволяет запретить включение в индекс

записей, имеющих значения Null в индексированных полях; зарезервированное слово PRIMARY позволяет назначить индексированные поля ключом. Такой индекс по умолчанию является уникальным, следовательно, зарезервированное слово UNIQUE можно опустить. Удаление таблицы/индекса Инструкция DROP удаляет существующую таблицу из базы данных или удаляет существующий индекс из таблицы. Синтаксис: DROP {TABLE таблица | INDEX индекс ON таблица} где таблица — имя таблицы, которую следует удалить или из которой следует удалить индекс; индекс — имя индекса, удаляемого из таблицы. Прежде чем удалить таблицу или удалить из нее индекс, необходимо ее закрыть. Следует отметить, что таблица удаляется из базы данных безвозвратно. Удаление записей Инструкция DELETE создает запрос на удаление записей из одной или нескольких таблиц, перечисленных в предложении FROM и удовлетворяющих предложению WHERE. Синтаксис команды: DELETE [Таблица.*] FROM таблица WHERE условие Отбора где Таблица — необязательное имя таблицы, из которой удаляются записи; таблица — имя таблицы, из которой удаляются записи; условие Отбора — выражение, определяющее удаляемые записи. С помощью инструкция DELETE можно осуществлять удаление большого количества записей. Данные из таблицы также можно удалить и с помощью инструкции DROP, однако при таком удалении теряется структура таблицы. Если же применить инструкцию DELETE, удаляются только данные. При этом сохраняются структура таблицы и все остальные ее свойства, такие, как атрибуты полей и индексы. Запрос на удаление удаляет записи целиком, а не только содержимое указанных полей. Нельзя восстановить записи, удаленные с помощью запроса на удаление. Чтобы узнать, какие записи будут удалены, необходимо посмотреть результаты запроса на выборку, использующего те же самые условие отбора в предложении Where, а затем выполнить запрос на удаление. Добавление записей Инструкция INSERT INTO добавляет запись или записи в таблицу. Синтаксис команды: а) запрос на добавление нескольких записей: INSERT INTO назначение [(поле_1[, поле_2[, ...]])] SELECT [источник.]поле_1[, поле_2[, ...] FROM выражение б) запрос на добавление одной записи: INSERT INTO назначение [(поле_1[, поле_2[, ...]])] VALUES (значение_1[, значение_2[, ...]) где назначение — имя таблицы или запроса, в который добавляются записи; источник — имя таблицы или запроса, откуда копируются записи; поле_1, поле_2 — имена полей для добавления данных, если они следуют за аргументом «Назначение»; имена полей, из которых берутся данные, если они следуют за аргументом источник; выражение — имена таблицы или таблиц, откуда вставляются данные. Это выражение может быть именем отдельной таблицы или результатом операции INNER JOIN, LEFT JOIN или RIGHT JOIN, а также сохраненным запросом; значение_1, значение_2 — значения, добавляемые в указанные поля новой записи. Каждое значение будет вставлено в поле, занимающее то же положение в списке: значение_1 вставляется в поле_1 в новой записи, значение_2 — в поле_2 и т.д. Каждое значение текстового поля следует заключать в кавычки (' '), для разделения значений используются запятые. Инструкцию INSERT INTO можно использовать для добавления одной записи в таблицу с помощью запроса на добавление одной записи, описанного выше. В этом случае инструкция должна содержать имя и значение каждого поля записи. Нужно определить все поля записи, в которые будет помещено значение, и значения для этих полей. Если поля не определены, в недостающие столбцы будет вставлено значение по умолчанию или значение Null. Записи добавляются в конец таблицы. Инструкцию INSERT INTO можно также использовать для добавления набора записей из другой таблицы или запроса с помощью предложения SELECT ... FROM, как показано выше в запросе на добавление нескольких записей. В этом случае предложение SELECT определяет поля, добавляемые в указанную таблицу НАЗНАЧЕНИЕ. Инструкция INSERT INTO является необязательной, однако, если она присутствует, то должна находиться перед инструкцией SELECT. Запрос на добавление записей копирует записи из одной или нескольких таблиц в другую таблицу. Таблицы, которые содержат добавляемые записи, не изменяются. Вместо добавления существующих записей из другой таблицы, можно указать значения полей одной новой записи с помощью предложения VALUES. Если список полей опущен, предложение

VALUES должно содержать значение для каждого поля таблицы; в противном случае инструкция INSERT не будет выполнена. Можно использовать дополнительную инструкцию INSERT INTO с предложением VALUES для каждой добавляемой новой записи. Обновление данных Инструкция UPDATE создает запрос на обновление, который изменяет значения полей указанной таблицы на основе заданного условия отбора. Синтаксис команды: UPDATE таблица SET новое Значение WHERE условие Отбора; где таблица — имя таблицы, данные в которой следует изменить; новое Значение — выражение, определяющее значение, которое должно быть вставлено в указанное поле обновленных записей; условие Отбора — выражение, отбирающее записи, которые должны быть изменены. При выполнении этой инструкции будут изменены только записи, удовлетворяющие указанному условию. Инструкцию UPDATE особенно удобно использовать для изменения сразу нескольких записей или в том случае, если записи, подлежащие изменению, находятся в разных таблицах. Одновременно можно изменить значения нескольких полей. Следующая инструкция SQL увеличивает стипендию студентов группы 422-1 на 10 %: UPDATE Студенты SET Стипендия = стипендия * 1.1 WHERE Номер_группы = '422-1'; Запрос на объединение Операция UNION создает запрос на объединение, который объединяет результаты нескольких независимых запросов или таблиц. Синтаксис команды: [TABLE] запрос_1 UNION [ALL] [TABLE] запрос_2 [UNION[ALL] [TABLE] запрос_n [...]] где запрос_1-n — инструкция SELECT или имя сохраненной таблицы, перед которым стоит зарезервированное слово TABLE. В одной операции UNION можно объединить в любом наборе результаты нескольких запросов, таблиц и инструкций SELECT. В следующем примере объединяется существующая таблица СТУДЕНТЫ и инструкции SELECT: TABLE Студенты UNION ALL SELECT * FROM Абитуриенты WHERE Общий_балл > 22; По умолчанию повторяющиеся записи не возвращаются при использовании операции UNION, однако в нее можно добавить предикат ALL, чтобы гарантировать возврат всех записей. Кроме того, такие запросы выполняются несколько быстрее. Все запросы, включенные в операцию UNION, должны отбирать одинаковое число полей; при этом типы данных и размеры полей не обязаны совпадать. Псевдонимы необходимо использовать только в первом предложении SELECT, в остальных они пропускаются. В каждом аргументе «Запрос» допускается использование предложения GROUP BY или HAVING для группировки возвращаемых данных. В конец последнего аргумента «Запрос» можно включить предложение ORDER BY, чтобы отсортировать возвращенные данные. Основные различия Microsoft Jet SQL и ANSI SQL Рассмотрим различия двух диалектов языка SQL Microsoft Jet SQL и ANSI SQL, описанные в руководстве разработчика приложений баз данных на СУБД Microsoft Access: языки SQL-ядра базы данных Microsoft Jet и ANSI SQL имеют разные наборы зарезервированных слов и типов данных; разные правила применимы к оператору Between...And, имеющему следующий синтаксис: выражение [NOT] Between значение_1 And значение_2. В языке SQL Microsoft Jet значение_1 может превышать значение_2; в ANSI SQL, значение_1 должно быть меньше или равно значение_2; разные подстановочные знаки используются с оператором Like. Так, в языке SQL Microsoft Jet любой одиночный символ изображается знаком «?», а в ANSI SQL знаком «_», любое число символов в языке SQL Microsoft Jet изображается знаком «*», а в ANSI SQL — знаком «%»; язык SQL-ядра Microsoft Jet обычно предоставляет пользователю большую свободу. Например, разрешается группировка и сортировка по выражениям. Особые средства языка SQL Microsoft Jet В языке SQL Microsoft Jet поддерживаются следующие дополнительные средства: инструкция TRANSFORM, предназначенная для создания перекрестных запросов; дополнительные статистические функции, такие как StDev и VarP; описание PARAMETERS, предназначенное для создания запросов с параметрами. Средства ANSI SQL, не поддерживаемые в языке SQL Microsoft Jet В языке SQL Microsoft Jet не поддерживаются следующие средства ANSI SQL: инструкции, управляющие защитой, такие как COMMIT, GRANT и LOCK; зарезервированное слово DISTINCT в качестве описания аргумента статистической функции (например, нельзя использовать выражение SUM(DISTINCT имя

Столбца); предложение LIMIT TO nn ROWS, используемое для ограничения количества строк, возвращаемых в результате выполнения запроса. Для ограничения количества возвращаемых запросом строк можно использовать только предложение WHERE. Задание Все запросы, создаваемые в рамках данной лабораторной работы, должны быть реализованы на языке SQL без помощи построителя запросов. Для создания нового SQLзапроса в окне базы данных нажмите кнопку Создание запроса в режиме конструктора и не выбирая таблицы для запроса нажмите кнопку Вид на панели инструментов. 1. Используя инструкцию CREATE TABLE, создайте запрос на создание новой таблицы для выбранной ранее предметной области, 20 содержащую пять полей, различных типов данных, определив в запросе первичный ключ и проиндексировав соответствующие поля, используя предложение CONSTRAINT. Для запуска запроса нажмите кнопку Запуск на панели инструментов. После чего создайте запрос на создание еще одной таблицы, содержащей внешний ключ по отношению к первичному ключу предыдущей таблицы. Запустите запрос, после чего проверьте, отразились ли изменения в схеме данных. 2. Используя команду SELECT, создайте запрос на выборку записей из двух (или более) таблиц, используя правила внешнего и внутреннего соединения, а также различные условия отбора и сортировки. 3. Используя команду UPDATE, создайте запрос на обновление данных в созданных ранее таблицах. 4. Используя команду INSERT INTO, создайте запросы на добавление группы записей и одной записи в существующую таблицу. 5. Используя команду CREATE INDEX, создайте запрос на создание нового индекса, используя различные условия на значения индексов (IGNORE NULL, DISSALLOW NULL, PRIMARY), а также типы сортировки. 6. Используя команду DROP, создайте запросы на удаление таблицы и индекса, созданных ранее в БД. 7. Сохраните все созданные запросы в базе данных.

Лабораторная работа № 8.

Тема: Создание концептуальной модели данных в среде автоматизированного проектирования.

Цель работы: Моделирование данных с помощью ER-диаграмм.

Теоретическая часть

Каждая сущность является множеством подобных индивидуальных объектов, называемых экземплярами. Каждый экземпляр индивидуален и должен отличаться от всех остальных экземпляров. Атрибут выражает определенное свойство объекта. С точки зрения БД (физическая модель) сущности соответствует таблица, экземпляру сущности – строка в таблице, а атрибуту – колонка таблицы. Построение модели данных предполагает определение сущностей и атрибутов, т.е. необходимо определить, какая информация будет храниться в конкретной сущности или атрибуте. Сущность можно определить как объект, событие или концепцию, информация о которой должна сохраняться. Сущности должны иметь наименование с четким смысловым значением, именоваться существительным в единственном лице, не носить «технических» наименований и быть достаточно важными для того, чтобы их моделировать. Именование сущности в единственном числе облегчает в дальнейшем чтение модели. Фактически имя сущности дается по имени ее экземпляра. Каждая сущность должна быть полностью определена с помощью текстового описания. Каждый атрибут хранит информацию об определенном свойстве сущности, а каждый экземпляр сущности должен быть уникальным. Атрибут или группа атрибутов, которые идентифицируют сущность, называется первичным ключом. При установлении связей между сущностями атрибуты первичного ключа родительской сущности мигрируют в качестве внешних ключей в дочернюю сущность. Очень важно дать атрибуту правильное имя. Атрибуты должны именоваться в единственном числе и иметь четкое смысловое значение. Соблюдение этого правила позволяет частично решить проблему нормализации данных уже на этапе определения атрибутов. Связи Связь является логическим соотношением между

сущностями. Каждая связь должна именоваться глаголом или глагольной фразой. Имя связи выражает некоторое ограничение или бизнес-правило и облегчает чтение построенной модели данных. Различают зависимые и независимые сущности. Тип сущности определяется ее связью с другими сущностями. Идентифицирующая связь устанавливается между независимой (родительский конец связи) и зависимой (дочерний конец связи) сущностями. При установлении идентифицирующей связи атрибуты первичного ключа родительской сущности переносятся в состав первичного ключа дочерней сущности. Эта операция дополнения атрибутов дочерней сущности при создании связи называется миграцией атрибутов. В дочерней сущности атрибуты помечаются как внешний ключ (FK). При установлении неидентифицирующей связи дочерняя сущность остается независимой, а атрибуты первичного ключа родительской сущности мигрируют в состав неключевых компонентов родительской сущности. Неидентифицирующая связь служит для связывания независимых сущностей. Имя связи – фраза, характеризующая отношение между родительской и дочерней сущностями. Для связи один-ко-многим идентифицирующей или не идентифицирующей достаточно указать имя, характеризующее отношение от родительской к дочерней сущности. Тип связи (идентифицирующая/неидентифицирующая). Для неидентифицирующей связи можно указать обязательность. В случае обязательной связи атрибут внешнего ключа получит признак NOT NULL, несмотря на то, что внешний ключ не войдет в состав первичного ключа дочерней сущности. В случае необязательной связи внешний ключ может принимать значение NULL. Необязательная неидентифицирующая связь помечается прозрачным ромбиком со стороны родительской сущности. Правила ссылочной целостности – логические конструкции, которые выражают бизнес-правила использования данных и представляют собой правила вставки, замены и удаления. Информацию о предметной области суммируют составлением спецификаций по сущностям, атрибутам и отношениям с использованием графических диаграмм, в чем и заключается процесс моделирование данных. Основы работы в Power Designer Для запуска пакета Power Designer в меню программы (Windows) найдите папку Sybase и запустите файл Power Designer. Для создания концептуальной модели данных необходимо выбрать File/ New или на панели инструментов выбрать значок . Далее появится окно для выбора создаваемой модели (рис. 7), в котором надо выбрать Conceptual Data Model

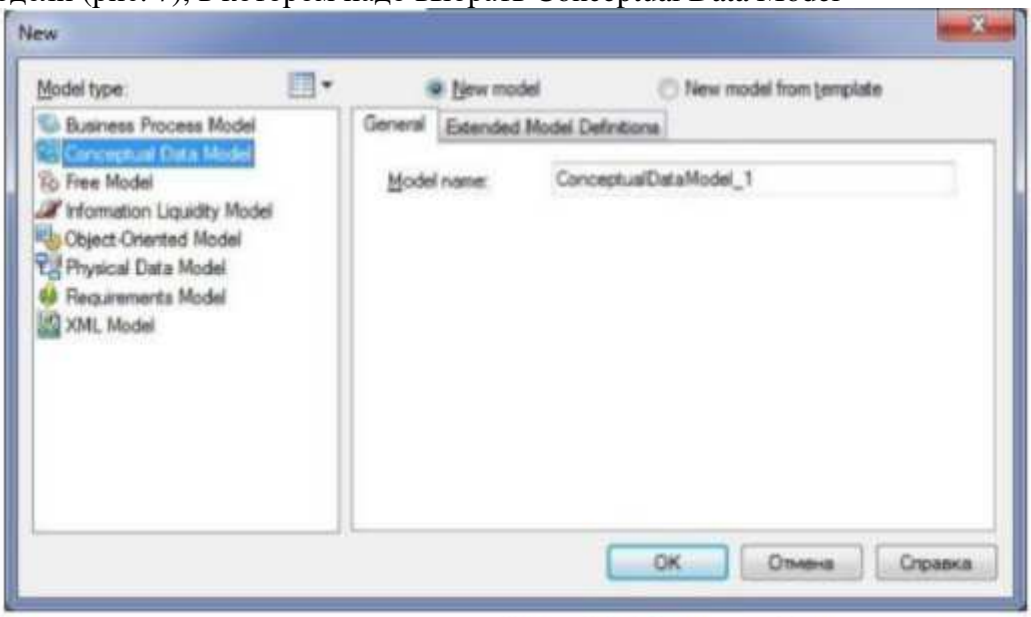


Рис. 7. Окно выбора модели.

После нажатия кнопки ОК появиться окно, в котором создается ER-диаграмма. Создание сущностей Для создания сущности, в панели Palette (рис. 8) нажмите кнопку с белым прямоугольником (с подсказкой Entity).

Рис. 8. Панель элементов с выбранным элементом сущность

Далее, поместите указатель мыши на рабочее поле в нужном месте и щелкните кнопкой мыши. Прямоугольник, изображающий сущность появится в указанном месте. При этом, курсор мыши на рабочем поле выглядит как выбранный элемент, т.о. можно создавать несколько выбранных элементов одного типа без повторного их выбора на панели элементов. Для того, чтобы изменить свойства созданной сущности, дважды щелкните на нее левой кнопкой мыши или нажмите правую кнопку и в выпавшем меню, выберите пункт Properties, в результате чего откроется окно свойств сущности (рис. 9).

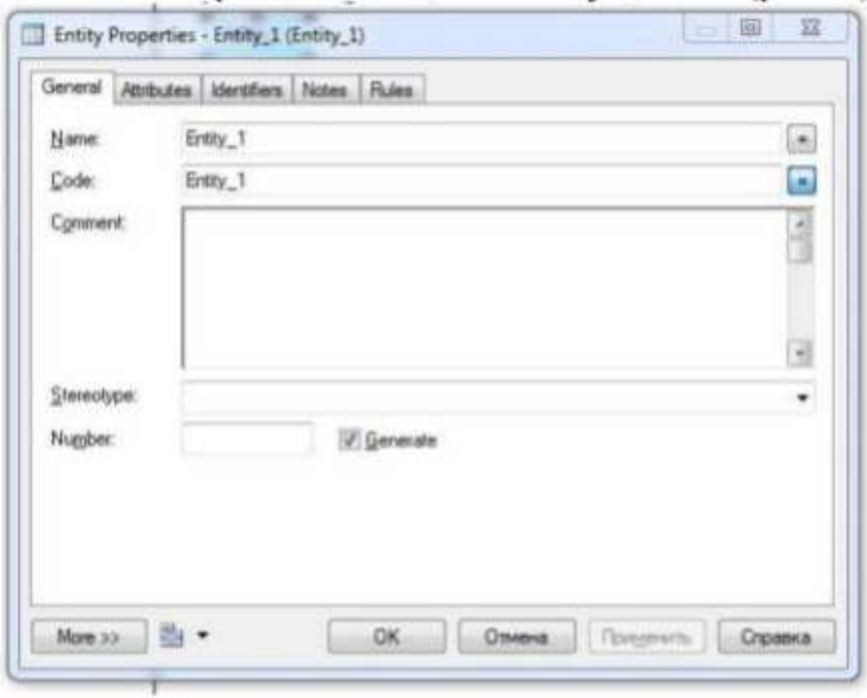


Рис. 9. Окно свойств сущности

В открывшемся окне пять закладок. Закладка General позволяет ввести следующие основные параметры: Name — имя сущности, которое будет видеть пользователь; Code — имя кода сущности, которое будет использоваться при генерации физической модели; Number — ограничение количества записей в таблице после генерации физической модели; Comment — комментарий, предназначенный для улучшения понимания модели.

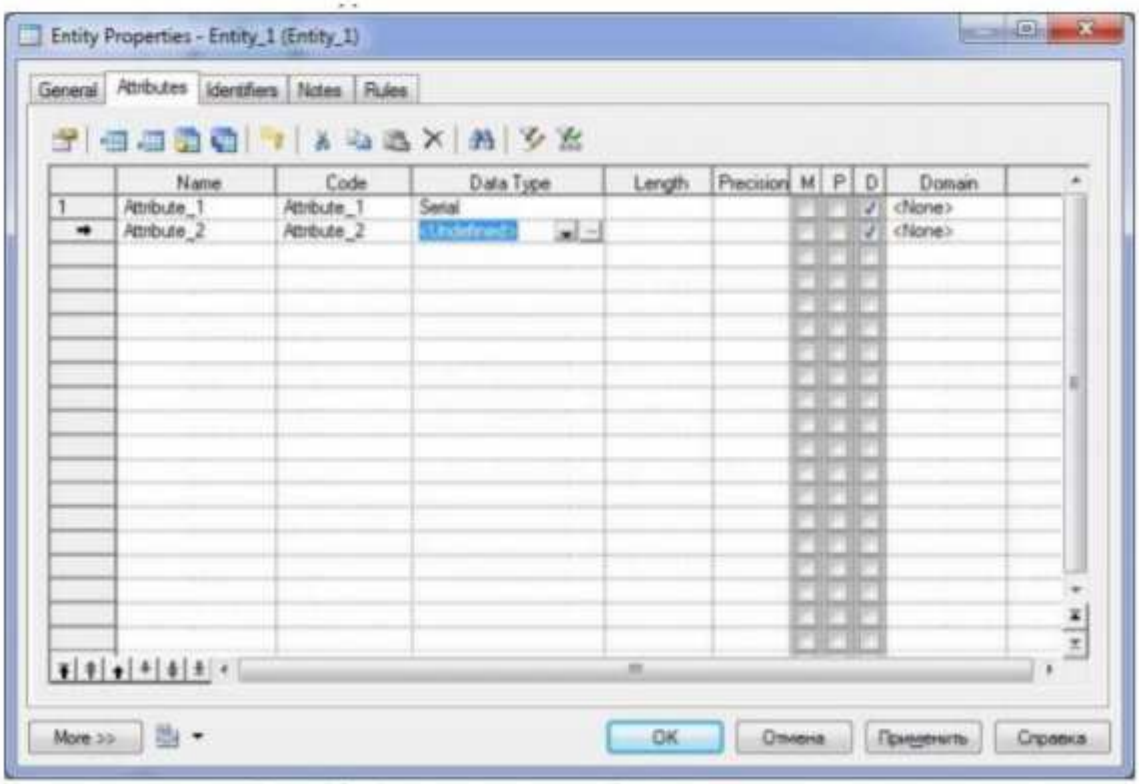


Рис. 10. Окно ввода атрибутов сущностей

Закладка Attributes содержит таблицу (рис. 10) и позволяет определять атрибуты сущности: Name — имя атрибута, которое будет видеть пользователь; Code — имя кода атрибута, которое будет использоваться при генерации физической модели; Data type — тип данных атрибута, который может быть выбран из выпадающего списка или вручную, щелкнув в поле Data type; Domain — принадлежность к домену, если он определен. Использование доменов позволяет, определив один раз пользовательский тип данных, использовать его в дальнейшем при определении типа данных атрибута. О создании домена будет сказано ниже; M (mandatory) — обязательный атрибут, указывает может ли данный атрибут принимать неопределенные значения (обязательно ли данное поле для заполнения в таблице БД); P (Primary Identifier) — первичный идентификатор сущности (в физической модели данных атрибут будет являться первичным ключом или его составной частью); D (Displayed) — отображаемый, т.е. будет ли атрибут показываться в модели. Более полную информацию по свойствам атрибута можно получить, дважды щелкнув по полю, расположенному слева от поля с именем атрибута. Здесь можно вводить комментарий в поле Comment, задавать список значений для данного атрибута, определять верхние и нижние границы значений Закладка Identifiers содержит автоматически заполняемую таблицу первичных идентификаторов сущности, но позволяет делать это вручную, когда необходимо создать суррогатный первичный идентификатор. Закладка Rules позволяет вводить необходимые правила на ввод значений в таблицу. Остальные закладки Notes, Version info носят описательный характер для улучшения понимания модели. Для фиксации всех изменений в необходимо нажать кнопку Apply. Домен Домен – это множество допустимых значений атрибута определенного типа данных. Домен определяется заданием стандартного типа данных, к которому относятся элементы домена и заданием произвольного логического выражения, применяемому к этому типу данных. Для создания домена необходимо в главном меню выбрав пункт Model, выбрать пункт Domains. На рис. 11 показано форма определения домена. Данная форма содержит аналогичные поля как для определения свойств атрибутов (в закладке Attributes в свойствах формы), а также дополнительные поля Length и Precision для описания длины и точности значений атрибутов.

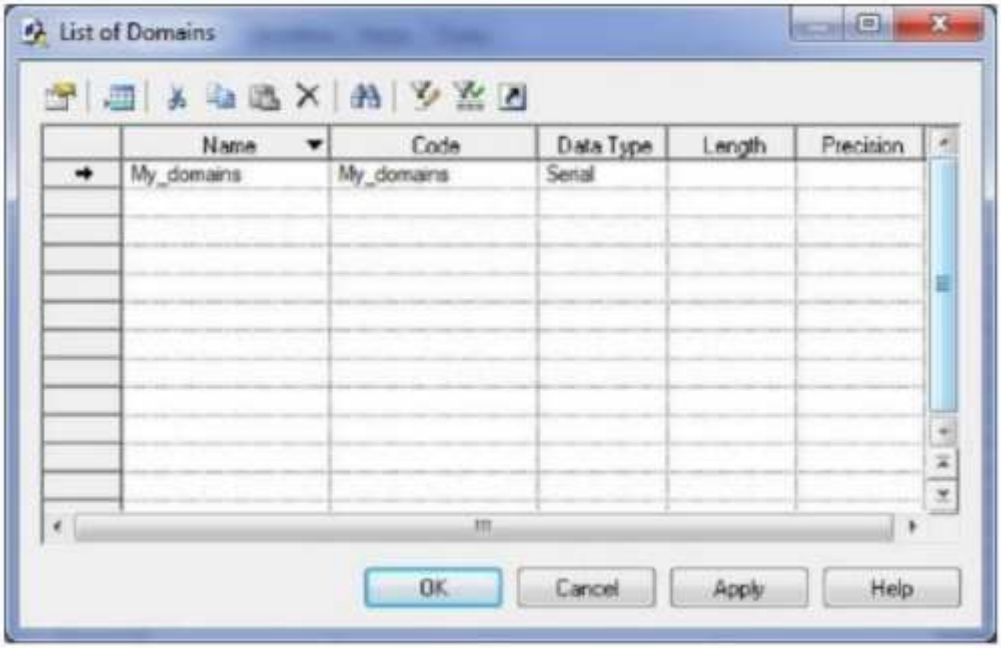


Рис. 11. Окно ввода списка доменов

После создания нового домена его имя появится в списке доменов при задании свойств атрибутов сущностей (рис. 12).

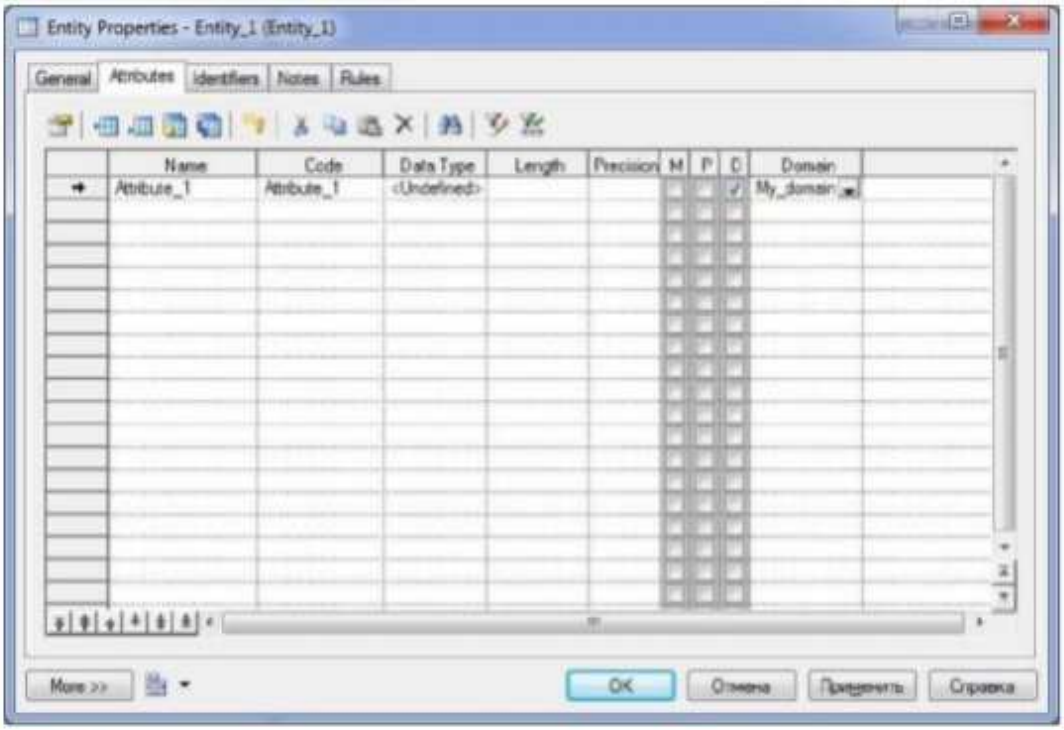


Рис. 12. Определение домена для атрибута сущности

При создании атрибутов сущностей в концептуальной модели не создаются атрибуты, являющиеся внешними ключами сущностей. После того, как созданы все необходимые сущности и атрибуты, необходимо определить связи между ними. Установка связей Для установки связи между двумя сущностями, необходимо нажать кнопку (с двумя белыми прямоугольниками и линией между ними рис. 13).

Рис. 13. Панель элементов с выбранным элементом связь

Необходимо перевести курсор мыши на одну сущность и, нажав левую кнопку мыши и, не отпуская ее, перевести курсор на вторую сущность. Далее можно отпустить кнопку мыши – связь установлена. Для изменения свойств связи, необходимо дважды щелкнуть левой кнопкой мыши на линию связи (или нажать правую кнопку мыши и выпавшем меню выбрать пункт Properties). Откроется окно (рис. 14), с закладками: закладки Notes и Version info используются для подробного описания связи. В закладке Rules можно задавать параметры ограничения связи; в закладке General указывается имя и код связи, а две кнопки с именами используемых сущностей позволяют вызвать окно со свойствами соответствующей сущности. Закладка Cardinalites позволяет указать вид связи (один– к–одному, один–ко–многим, многие–ко–многим и т.д.) и устанавливает свойства связи от Сущности1 к Сущности2 и наоборот: Mandatory определяет обязательность связи, показывая, что экземпляр Сущности1 (запись) может существовать только при наличии соответствующего экземпляра в Сущности2;

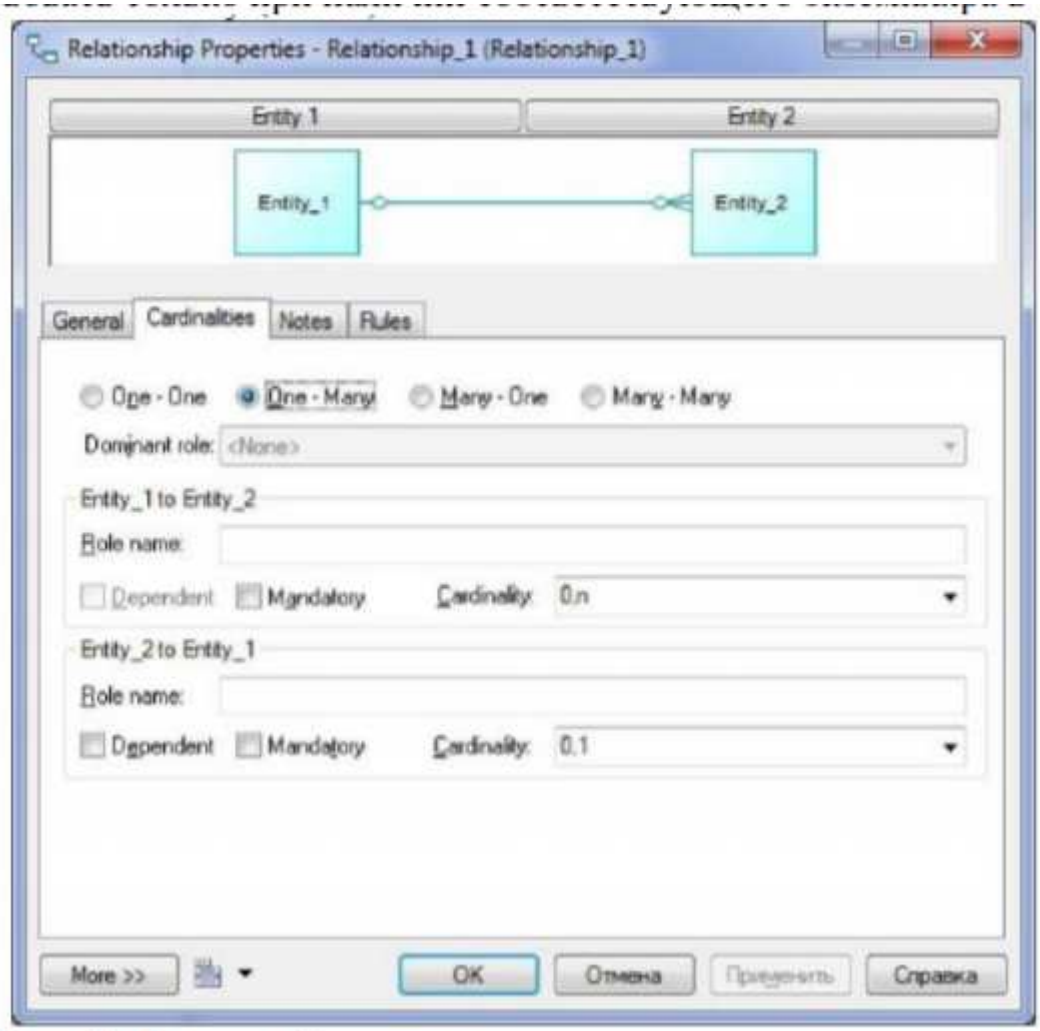


Рис. 14. Окно свойств связи

Dependent показывает, что каждый экземпляр Сущности1 отождествляется с экземпляром в Сущности2 (первичный ключ на стороне «один» при создании физической модели войдет в состав первичного ключа в таблице на стороне «многие»); Role – текст, описывающий связь от Сущности1 к Сущности2. Связи многие-ко-многим преобразуются в физической модели в промежуточные таблицы. После того, как созданы все сущности, указаны атрибуты и установлены все связи необходимо проверить концептуальную правильность построения концептуальной модели. Для этого необходимо выбрать в меню Tools/Check Model (или нажать F4). Появится окно (рис. 15), в котором предлагается выбрать объекты для проверки. Package – система проверит правильность циклических связей. Domain – система проверит правильность заполнения доменов. Data items – проверять ли атрибуты. Entities – система проверит правильность создания сущностей. Entity attributes – проверка правильности свойств сущности Entity identifier - проверка правильности идентификаторов сущности. Relationships – проверка связей.

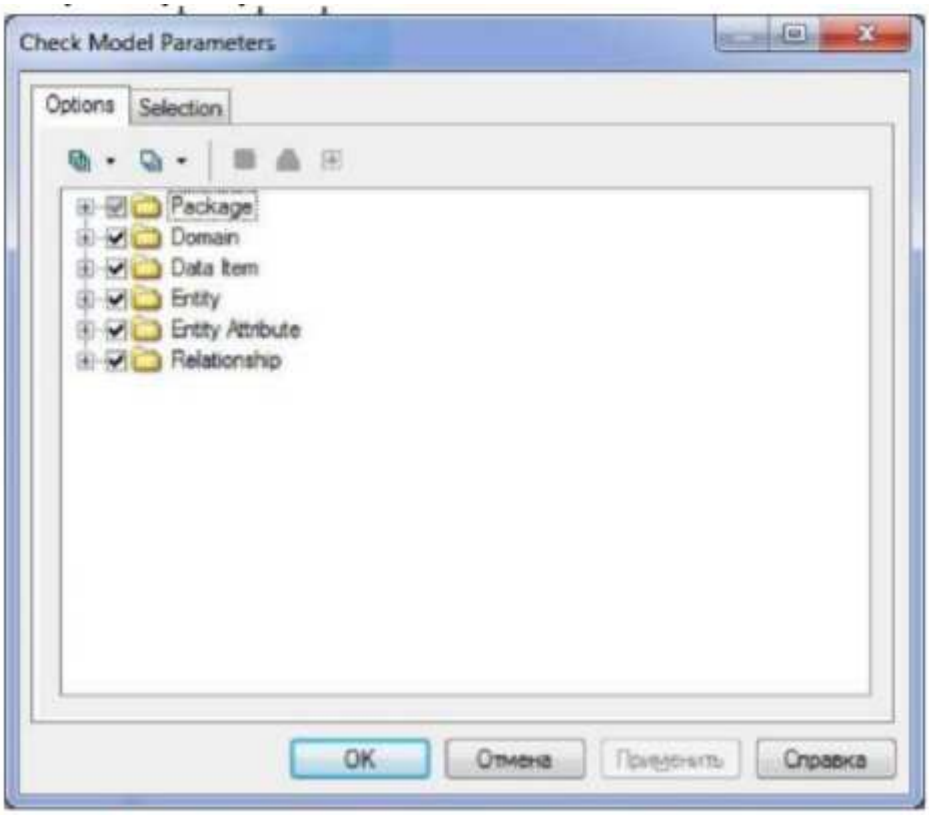


Рис. 15. Окно проверки концептуальной модели

После нажатия кнопки ОК система проверит всю концептуальную модель, выдаст ошибки (или предупреждения), если таковые имеются. Для просмотра сведений об ошибке необходимо дважды «щелкнуть» по ней кнопкой мыши. Задание 1.. Создайте концептуальную модель выбранной предметной области, учитывая следующие требования: a. Должно быть создано не менее 8-ми сущностей. b. В каждой сущности (за исключением справочников-классификаторов) должно быть создано не менее 5-ти атрибутов. c. Необходимо использовать домены для определения типов данных атрибутов. d. Обязательно соблюдение 3-й нормальной формы для проектируемых сущностей. e. Имена полей должны быть представлены по-русски, коды полей должны быть написаны латиницей; f. В концептуальной модели данных не должно быть создано внешних ключей; g. Необходимо корректно задать имена связей. 2. Выполните проверку качества созданной модели. Работа считается полностью выполненной, если при проверке модели не выдаются ошибки.