МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ

«БРАТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Базовая кафедра менеджмента и информационных технологий

УТВЕРЖДАЮ:
Проректор по учебной работе
_____Е.И. Луковникова
«_____» декабря 2018 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

БАЗЫ ДАННЫХ

Б1.Б.18

НАПРАВЛЕНИЕ ПОДГОТОВКИ

09.03.03 Прикладная информатика

ПРОФИЛЬ ПОДГОТОВКИ

Прикладная информатика в экономике

Программа академического бакалавриата

Квалификация (степень) выпускника: бакалавр

СОДЕРЖАНИЕ ПРОГРАММЫ **Стр.**

1. ПЕРЕЧЕНЬ ПЛАНИРУЕМЫХ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ, СООТНЕСЕННЫХ С ПЛАНИРУЕМЫМИ РЕЗУЛЬТАТАМИ ОСВОЕНИЯ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Вид деятельности выпускника

Дисциплина охватывает круг вопросов, относящихся к проектной, производственно-технологической видам профессиональной деятельности выпускника в соответствии с компетенциями и видами деятельности, указанными в учебном плане.

Цель дисциплины

овладение обучающимися основами теоретических и практических знаний в области баз данных, современных инструментальных средств моделирования и управления доступом к информационным массивам.

Основными задачами дисциплины являются:

изучение теоретических и практических основ применения возможностей баз данных (проектирование, ведение и использование баз данных).

Код компетенции	Содержание компетенций	Перечень планируемых результатов обучения по дисциплине
1	2	3
ОПК-1	способностью использовать нормативно-правовые документы, международные и отечественные стандарты в области информационных систем и технологий	**знать:** – нормативно-правовые документы, международные и отечественные стандарты в области информационных систем и технологий **уметь:** – использовать нормативно-правовые документы, международные и отечественные стандарты в области информационных систем и технологий.; **владеть:** – навыками работы с нормативно-правовыми документами, международными и отечественными стандартами в области информационных систем и технологий
ПК-6	способностью собирать детальную информацию для формализации требований пользователей заказчика	**знать:** – основные направления сбора и анализа требований пользователей; **уметь:** – разрабатывать концептуальную модель прикладной области; – проводить формализацию и реализацию решения прикладных задач; **владеть:** – навыками построения концептуальной модели предметной области;
ПК-7	способностью проводить описание прикладных процессов и информационного обеспечения решения прикладных задач	**знать:** – основные конструкции информационной модели предметной области базы данных; – основные конструкции функциональной модели предметной области, предназначенных для описания процессов обработки информации, **уметь:** – использовать методологию моделирования

		«сущность-связь»; – использовать методологию IDEF0; **владеть:** – навыками ER-моделирования; – навыками работы с инструментальными средствами проектирования баз данных;
ПК-14	способность осуществлять ведение базы данных и поддержку информационного обеспечения решения прикладных задач	**знать:** – классификацию и модели данных; – основные положений концепции баз данных и принципов построения баз данных; – системы управления БД; – основные предложения языка запросов SQL; **уметь:** – выбирать инструментальные средства и технологии проектирования БД. – применять методы проектирования баз данных, – работать в конкретной СУБД; – работать с базой данных средствами языка SQL **владеть:** – навыками применения современных методов сбора, обработки и анализа данных; – навыками реализации проектирования реляционной БД, – навыками работы в конкретной СУБД

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Дисциплина Б1.Б.18 Базы данных относится к базовой.

Дисциплина Базы данных базируется на знаниях, полученных при изучении дисциплин: Информатика и программирование, Дискретная математика, Информационные системы и технологии.

Базы данных представляет основу для изучения дисциплин: Проектирование информационных систем; Разработка программных приложений; Программная инженерия.

Такое системное междисциплинарное изучение направлено на достижение требуемого ФГОС уровня подготовки по квалификации бакалавр.

3. РАСПРЕДЕЛЕНИЕ ОБЪЕМА ДИСЦИПЛИНЫ

3.1. Распределение объема дисциплины по формам обучения

Форма обучения	*Курс*	*Семестр*	*Трудоемкость дисциплины в часах*						*Курсовая работа*	*Вид промежуточной аттестации*
			Всего часов (экз)	*Аудиторных часов*	*Лекции*	*Лабораторные работы*	*Практические занятия*	*Самостоятельная работа*		
1	2	3	4	5	6	7	8	9	10	11
Очная	2	3,4	216	88	35	53	-	101	КР	зачет, экзамен
Заочная	3	-	216	25	9	-	16	182	КР	экзамен
Заочная (ускоренное обучение)	-	-	-	-	-	-	-	-	-	-
Очно-заочная	-	-	-	-	-	-	-	-	-	-

3.2. Распределение объема дисциплины по видам учебных занятий и трудоемкости

Вид учебных занятий	*Трудо-емкость (час.)*	*в т.ч. в интерактивной, активной, инновационной формах, (час.)*	*Распределение по семестрам, час*	
			3	*4*
1	2	3	4	5
I. Контактная работа обучающихся с преподавателем (всего)	88	30	34	54
Лекции (Лк)	35	12	17	18
Лабораторные работы (ЛР)	53	18	17	36
Курсовая работа	+	-	+	-
Групповые консультации	+	-	+	+
II.Самостоятельная работа обучающихся (СР)	101	-	2	99

Подготовка к лабораторным занятиям	72	-	-	72
Выполнение курсовой работы	2	-	2	-
Подготовка к экзамену в течение семестра	27	-	-	27
III. Промежуточная аттестация зачет	+	-	+	-
экзамен	27	-	-	27
Общая трудоемкость дисциплины . час.	216	-	36	180
зач. ед.	6	-	1	5

4. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

4.1. Распределение разделов дисциплины по видам учебных занятий

- для очной формы обучения:

№ раздела	*Наименование раздела*	*Трудоемкость, (час.)*	*Виды учебных занятий, включая самостоятельную работу обучающихся и трудоемкость; (час.)*		
			учебные занятия		*самостоятельная работа обучающихся*
			лекции	*Лабораторные работы*	
1	2	3	4	5	6
1.	**Основные положения теории БД.**	**8**	**4**	**0**	**4**
2.	**Модели данных.**	**6**	**2**	**0**	**4**
3.	**Реляционные модели и языки.**	**19**	**4**	**0**	**15**
4.	**Предметная область базы данных.**	**45**	**5**	**20**	**20**
5.	**Жизненный цикл БД.**	**15**	**4**	**0**	**11**
6.	**Проектирование БД.**	**65**	**11**	**27**	**27**
7.	**Структурированный язык запросов SQL.**	**31**	**5**	**6**	**20**
	ИТОГО	**189**	**35**	**53**	**101**

- для заочной формы обучения:

№ раздела	*Наименование раздела*	*Трудоемкость, (час.)*	*Виды учебных занятий, включая самостоятельную работу обучающихся и трудоемкость; (час.)*		
			учебные занятия		*самостоятельная работа обучающихся*
			лекции	*Лабораторные работы*	
1	2	3	4	5	6
1.	**Основные положения теории БД.**	**14,5**	**0,5**	**0**	**14**
2.	**Модели данных.**	**14,5**	**0,5**	**0**	**14**
3.	**Реляционные модели и языки.**	**25,5**	**0,5**	**0**	**25**
4.	**Предметная область базы данных.**	**22,5**	**0,5**	**0**	**22**
5.	**Жизненный цикл БД.**	**65**	**5**	**13**	**47**
6.	**Проектирование БД.**	**33**	**1**	**2**	**30**

7.	**Структурированный язык запросов SQL.**	**32**	**1**	**1**	**30**
	ИТОГО	**207**	**9**	**16**	**182**

4.2. Содержание дисциплины, структурированное по разделам и темам

Раздел 1. Основные положения теории БД. (комп. презентация – 4 часа)

Информация как социальный ресурс

Развитие систем связи и коммуникаций привело к усложнению и дифференциации информационных процессов в человеческом обществе. Способность накапливать информацию и обеспечивать эффективный доступ к ней становится определяющим фактором не только развития человеческого общества, но и поддержания его жизнеспособности. Быстрый рост объемов информации, закрепленной на внешних по отношению к человеку носителях, привел к появлению новых общественных институтов (библиотеки, архивы, пресса, вычислительные центры и т. д.) и специальных систем (службы научно-технической информации, справочные службы, глобальные информационные компьютерные сети).

Развитие средств вычислительной техники и информационных технологий открыло новые возможности и способы хранения, представления и поиска информации, в частности, создание вычислительных систем, "доступных по требованию" - т.е. вычислительные ресурсы становятся таким же доступным ресурсом для потребления человеком, как электроэнергия, природный газ, вода.

Таким образом, резко возрастают требования к качеству и надежности проектирования систем для работы с информацией, представляемой в электронном виде.

Информация и данные

Прежде чем перейти к обсуждению понятия информационной системы (ИС), попытаемся выяснить, что же понимается под словом информация. Ответить на этот вопрос и просто, и сложно: слово "информация" связано с широким кругом понятий, в том числе и определенных строго математически (информация по Шеннону, например).

Содержательная же сторона понятия "информация" очень многогранна и не имеет четких семантических границ. Однако всегда можно сказать, что можно с ней делать. Именно ответ на этот вопрос чаще всего и интересует как системных аналитиков и разработчиков (ИС), так и пользователей информации (ее основных потребителей).

С точки зрения как пользователей, так и разработчиков ИС, у информации есть одно важное свойство - она является единицей данных, подлежащих обработке. Обычно информация поступает потребителю именно в виде данных: таблиц, графиков, рисунков, фильмов, устных сообщений, которые фиксируют в себе информацию определенной структуры и типа. Таким образом, данные выступают как способ представления информации в определенной, фиксированной форме, пригодной для обработки, хранения и передачи. Хотя очень часто термины "информация" и "данные" выступают как синонимы (особенно в среде разработчиков ИС), следует помнить об этом их существенном отличии. Именно в данных информация обретает интерпретацию в конкретной ИС.

На рис. 1.2 просуммированы в общих чертах функции ИС через ее основные структурные компоненты.

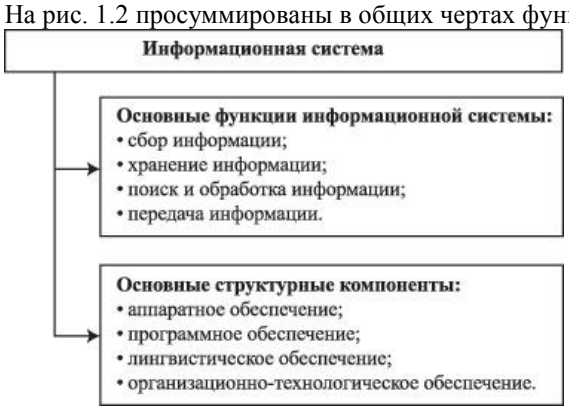


Рис. 1.2. Определение информационной системы

Итерационная процедура построения информационных систем

В чем состоит преимущество ориентации на автоматизацию основных бизнес-процессов при автоматизации организации или предприятия? Традиционно и повсеместно используемым подходом (особенно на начальных этапах развития информационной инфраструктуры организации) является применение так называемого позадачного метода решения задач автоматизации, направленного на решение достаточно простых и понятных руководству задач.

Низкая отдача уже существующей ИС организации на текущем этапе ее эксплуатации также становится тормозящим фактором. Изменение направлений бизнеса организации и ряд других факторов приводят к вопросу пересмотра отношения к ИС в организации, т.е. к извечному вопросу - переделать или начать с начала. Начать сначала всегда выгодней. Можно применить уже хорошо отработанные в информатике методики

проектирования "сверху-вниз" или "снизу-верх". Однако рано или поздно опять встанет вопрос о соответствии требованиям сегодняшнего дня.

Даже в тех случаях реализации ИС, которые одобряются системным анализом, не удается избежать переделки ИС, т.к. она, как органическая часть производственного процесса, должна следовать и отвечать стратегическому генеральному бизнес-плану развития организации. Такой план всегда должен быть, если организация собирается долго жить в своем секторе рынка.

Концепция баз данных

Подавляющее большинство компьютеров, которые используются для аппаратного обеспечения создателями ИС, являются компьютерами фон Неймана. Основная идея, положенная в основу создания компьютера фон Неймана, состоит в том, что компьютер представляет собой вычислительную машину с загружаемым в его память кодом - программами и данными. В процессе своей работы такая машина интерпретирует код и различает программы (исполняемый код) и данные (неисполняемый код).

Основные подходы к обработке информации в автоматизированных информационных системах

Одним из главных вопросов разработки программного обеспечения ИС (и программирования как самостоятельной дисциплины) является вопрос о соотнесении программ и данных, ибо решение этого вопроса, в конечном счете, определяет выбор алгоритмов обработки информации, аппаратных средств и технологической платформы. Фундаментальным принципом в решении вопроса о соотнесении программ и данных является концепция независимости прикладных программ от данных, и неважно, какая обработка данных предполагается: централизованная или распределенная. Суть этой концепции состоит не столько в отделении программ от данных, сколько в рассмотрении их как самостоятельных взаимодействующих объектов.

Одной из последних модификаций этого принципа является концепция независимости прикладных программ от данных вместе с процедурами их обработки (объектно-ориентированный подход в программировании), который позволяет решить ряд вопросов обработки данных, связанных с интерпретацией семантического смысла данных.

Основные концепции обработки данных

Концепция файловой системы

- данные ИС представляются в виде совокупности файлов несущей Операционной системы;
- структура файла определяется разработчиком ИС (совмещение логической и физической структуры данных);
- любая программа, содержащая запрос к данным, должна уметь интерпретировать структуру необходимых для реализации запроса файлов (зависимость программ обработки от организации данных);
- каждый запрос требует в принципе своей программы обработки (разнообразие точек зрения алгоритмическую обработку данных).

Концепция баз данных

- данные ИС размещаются в файлах несущей Операционной системы;
- физическая структура файлов фиксируется, а логическая структура данных представляется на ее основе (разделение и физической и логической структуры данных);
- обработка запроса к данным реализуется через специальный интерфейс манипулирования данными (независимость программ от организации данных);
- программы разрабатываются на основе единой точки зрения на процедуры обработки данных.

Концепция объектно-ориентированных баз данных

- данные ИС размещаются в файлах несущей Операционной системы;
- информация ИС представляется в виде объектов (логическая структура и методы - принцип разделения логической и физической структуры остается в силе);
- обработка запроса к данным реализуется через интерфейс манипулирования объектами (независимость программ и объектов);
- программы разрабатываются на основе единой точки зрения на классы объектов и идеи повторного использования программного кода.

Рис. 1.3. Основные концепции обработки информации

Методы обработки данных

Позадачный метод

Суть: декомпозиция программы на подзадачи о своими блоками данных, своими алгоритмами.
Проблемы:
- избыточность данных;
- взаимосвязь между данными и программой;
- замкнутость системы;
- нарушается естественная для учреждения схема обработки данных.

Метод баз данных

Суть: наличие отдельного описания логической структуры данных, единая точка зрения на процедуры обработки данных.
Проблемы:
- проблема представления данных;
- проблема обучения персонала;
- структурная реорганизация учреждения (реинжиниринг).

Метод объектно-ориентированных баз данных

Суть: информация (данные + программы) представляется объектами, которые обмениваются сообщениями (взаимосвязь процессов).
Проблемы:
- проблема определения объектов (классы);
- проблема обучения персонала;
- реорганизация учреждения.

Рис. 1.4. Основные методы обработки информации

Концепция баз данных

Идея повышения степени независимости обрабатывающих программ от способов хранения и содержания хранимых данных впервые была использована в концепции баз данных путем разделения логического и физического уровней хранения данных в 1964 году в исследованиях сотрудников фирмы IBM.

Что же принято понимать под базой данных? Базу данных в общем случае можно определить как унифицированную совокупность хранимых и воспроизводимых данных, используемых в рамках организации (Engles R.A., 1972 г.). Однако понятие базы данных не основывается в настоящее время на единой концепции, скорее это целое семейство связанных между собой понятий из предметной области, программного и аппаратного обеспечения, анализа и моделирования данных и приложений. Мы дадим несколько определений базы данных.

Определение 3. База данных.

(По Дж. Мартину) База данных есть совокупность взаимосвязанных данных, совместно используемых несколькими приложениями и хранящимися с (минимальной) регулируемой избыточностью. Данные запоминаются таким образом, чтобы они, по мере возможности, не зависели от программ. Для обработки данных применяется общий управляющий метод доступа. Если базы данных не пересекаются по структуре, то говорят о системе баз данных.

(Согласно материалам комитета КОДАСИЛ) База данных состоит из всех экземпляров записей, экземпляров наборов записей и областей, которые контролируются конкретной схемой. (Под схемой можно понимать карту всей логической структуры базы данных. Определение понятия схемы по КОДАСИЛ будет дано при обсуждении сетевой модели данных).

Такое деление средств манипулирования данными и их представления является в определенной степени условным. Язык определения данных служит для описания логической структуры (схемы) БД, а в некоторых случаях и способов хранения и доступа к данным. Язык манипулирования данными предоставляет алгоритмические средства построения приложений для обработки сохраняемых в БД элементов данных.

Системы управления базами данных

В ситуации применения концепции базы данных для создания ИС естественным образом возникает вопрос - а кто или что должно все это поддерживать? Таким образом, встает вопрос о Системе управления базой данных (СУБД). СУБД являются сложными программными системами, работающими на различных операционных платформах. Именно СУБД должна предоставить средства определения и манипулирования данными, сделав данные независимыми от прикладных программ, их использующих. В последнее время набирает обороты концепция машин баз данных, которая предполагает аппаратную реализацию некоторых процедур обработки данных.

Однако после признания концепции БД прошло почти четыре года, прежде чем в 1966 году была создана первая СУБД. На рис. 1.5представлены основные функции СУБД.

Основные функции СУБД
• обеспечить языковые средства описания и манипулирования данными; • обеспечить поддержку логических моделей данных; • обеспечить взаимодействие логической и физической структур данных; • обеспечить защиту и целостность данных; • обеспечить поддержку баз данных в актуальном состоянии.

Рис. 1.5. Основные функции СУБД

Определение 4. Системой управления базами данных (Data-base Management System) называется совокупность программных средств, необходимых для использования базы данных и предоставляющих разработчикам и пользователям множество различных представлений данных.

Классификация баз данных

По технологии обработки данных базы данных подразделяются на централизованные и распределенные.

Централизованная база данных хранится в памяти одной вычислительной системы. Эта *вычислительная система* может быть мэйнфреймом - тогда *доступ* к ней организуется с использованием терминалов - или файловым сервером локальной сети ПК.

Распределенная база данных состоит из нескольких, возможно, пересекающихся или даже дублирующих друг друга частей, которые хранятся в различных ЭВМ вычислительной сети. Работа с такой базой осуществляется с помощью *системы управления распределенной базой данных* (*СУБД*).

По способу доступа к данным *базы данных* разделяются на **базы данных с локальным доступом** и **базы данных с сетевым доступом.**

Для всех современных баз данных можно организовать сетевой *доступ* с многопользовательским режимом работы.

Централизованные *базы данных* с сетевым доступом могут иметь следующую архитектуру:

- *файл-сервер* ;
- *клиент-сервер* базы данных;
- "тонкий клиент" - сервер приложений - сервер базы данных (трехуровневая архитектура).

Рис. 2.1. Схема работы с БД в локальной сети с выделенным файловым сервером

Файл-сервер. *Архитектура* систем *БД* с сетевым доступом предполагает выделение одной из машин сети в качестве центральной (*файловый сервер*). На этот *компьютер* устанавливается *операционная система* (ОС) для выделенного сервера (например, Microsoft *Windows Server* 2003). На нем же хранится совместно используемая централизованная *БД* в виде

одного или группы файлов. Все другие компьютеры сети выполняют функции рабочих станций (могут работать в ОС Microsoft *Windows* 2000 *Professional* или Microsoft *Windows* 98). Файлы *базы данных* в соответствии с пользовательскими запросами передаются на рабочие станции, где и производится обработка информации (см. рис. 2.1). При большой интенсивности доступа к одним и тем же данным *производительность* информационной системы падает. Пользователи могут создавать также *локальные БД* на рабочих станциях.

Рис. 2.2. Схема работы с БД в архитектуре "Клиент-сервер"

Клиент-сервер. В этой архитектуре на выделенном сервере, работающем под управлением серверной операционной системы, устанавливается специальное *программное обеспечение* (*ПО*) - *сервер БД*, например, Microsoft ® *SQL Server* ™ или *Oracle*. *СУБД* подразделяется на две части: клиентскую и серверную. Основа работы сервера *БД* - использование языка запросов (*SQL*). *Запрос* на языке *SQL*, передаваемый клиентом (рабочей станцией) серверу *БД*, порождает *поиск* и *извлечение данных* на сервере. Извлеченные данные транспортируются *по* сети от сервера к клиенту (см. рис. 2.2). Тем самым, количество передаваемой *по* сети информации уменьшается во много раз.

Трехуровневая архитектура функционирует в Интранет- и Интернет-сетях. Клиентская часть ("тонкий клиент"), взаимодействующая с пользователем, представляет собой HTML-страницу в Web-браузере либо *Windows-приложение*, взаимодействующее с Web-сервисами. Вся программная логика вынесена на *сервер* приложений, который обеспечивает формирование запросов к базе данных, передаваемых на выполнение серверу баз данных. *Сервер* приложений может быть Web-сервером или специализированной программой (например, *Oracle* Forms*Server*) (см. рис. 2.3).

Рис. 2.3. Схема работы с БД в трехуровневой архитектуре

Раздел 2. Модели данных.

Описание: Иерархическая модель данных. Сетевая модель данных. Объектно-ориентированная модель данных. Постреляционная модель данных.

Модель данных

Модель данных - интегрированный набор понятий для описания и обработки данных, связей между ними и ограничений, накладываемых на данные в некоторой организации.

Модель является представлением "реального мира" объектов и событий, а также существующих между ними связей. Это некоторая абстракция, в которой акцент делается на самых важных и неотъемлемых аспектах деятельности организации, а все второстепенные свойства игнорируются. Таким образом, можно сказать, что модель данных представляет саму организацию. Модель должна отражать основные концепции, представленные в таком виде, который позволит проектировщикам и пользователям базы данных обмениваться конкретными и недвусмысленными мнениями о роли тех или иных данных в организации. Модель данных можно рассматривать как сочетание трех указанных ниже компонентов.

- Структурная часть, т.е. набор правил, по которым может быть построена база данных.
- Управляющая часть, определяющая типы допустимых операций с данными (сюда относятся операции обновления и извлечения данных, а также опе рации изменения структуры базы данных).
- Набор (необязательный) ограничений поддержки целостности данных, гарантирующих корректность используемых данных.

Цель построения модели данных заключается в представлении данных в понятном виде. Если такое представление возможно, то модель данных можно легко применить при проектировании базы данных.

Модели данных подразделяются на три категории:

- объектные (object-based) модели данных,
- модели данных на основе записей (record-based),
- физические модели данных.

Первые две используются для описания данных на концептуальном и внешнем уровнях, а последняя — на внутреннем уровне.

Объектные модели данных

При создании объектных моделей данных используются следующие понятия:

- **Сущность** — это отдельный элемент деятельности организации (сотрудник или клиент, место или вещь, понятие или событие), который должен быть представлен в базе данных.
- **Атрибут** — это свойство, которое описывает некоторый аспект объекта и значение которого следует зафиксировать.
- **Связь** — это ассоциативное отношение между сущностями.

Ниже перечислены некоторые наиболее общие типы объектных моделей данных.

- Модель типа "сущность-связь", или ER-модель (Entity-Relationship model).

В настоящее время ER-модель стала одним из основных методов концептуального проектирования баз данных.

- Семантическая модель.
- Функциональная модель.
- Объектно-ориентированная модель.

Объектно-ориентированная модель расширяет определение сущности с целью включения в него не только атрибутов, которые описывают состояние объекта, но и действий, которые с ним связаны, т.е. его поведение. В таком случае говорят, что объект инкапсулирует состояние и поведение.

Модели данных на основе записей

В модели на основе записей база данных состоит из нескольких записей фиксированного формата, которые могут иметь разные типы. Каждый тип записи определяет фиксированное количество полей, каждое из которых имеет фиксированную длину.

Существуют три основных типа логических моделей данных на основе записей:

- реляционная модель данных (relational data model),
- сетевая модель данных (network data model),
- иерархическая модель данных (hierarchical data model).

Физические модели данных

Физические модели данных описывают то, как данные хранятся в компьютере, представляя информацию о структуре записей, их упорядоченности и существующих путях доступа. Физических моделей данных не так много, как логических, а самыми популярными среди них являются обобщающая модель (unifying model) и модель памяти кадров (frame memory).

Иерархическая модель данных

Представление связей в иерархической модели.

Иерархическая модель является ограниченным подтипом сетевой модели. В ней данные также представлены как коллекции записей, а связи — как наборы. Однако в иерархической модели узел может иметь только одного родителя. Иерархическая модель может быть представлена как древовидный граф с записями в виде узлов (которые также называются сегментами) и множествами в виде ребер.

Данная модель характеризуется такими параметрами, как уровни, узлы, связи. Принцип работы модели таков, что несколько узлов более низкого уровня соединяется при помощи связи с одним узлом более высокого уровня.

Также можно встретить следующее описание типов иерархической модели.

Тип «дерево» является составным. Он включает в себя подтипы («поддеревья»), каждый из которых, в свою очередь, является типом «дерево». Каждый из типов «дерево» состоит из одного «корневого» типа и упорядоченного набора (возможно пустого) подчиненных типов. Каждый из элементарных типов, включенных в тип «дерево», является простым или составным типом «запись». Простая «запись» состоит из одного типа, например, числового, а составная «запись» объединяет некоторую совокупность типов, например, целое, строку символов и указатель (ссылку).

К достоинствам иерархической модели данных относятся эффективное использование памяти ЭВМ и неплохие показатели времени выполнения основных операций над данными. Иерархическая модель данных удобна для работы с иерархически упорядоченной информацией. Недостатком иерархической модели является ее громоздкость для обработки информации с достаточно сложными логическими связями, а также сложность понимания для обычного пользователя.

Сетевая модель данных

Сетевая модель данных – модель, состоящая из записей, элементов данных и связей типа “один ко многим” (1:M), установленных между записями.

В сетевой модели данные представлены в виде коллекций записей, а связи - в виде наборов. В отличие от реляционной модели, связи здесь явным образом моделируются наборами, которые реализуются с помощью указателей. Сетевую модель можно представить как граф с записями в виде узлов графа и наборами в виде его ребер. Наиболее полно концепция сетевых БД впервые изложена в предложения группы CODASYL.

Для описания схемы сетевой БД используется две группы типов: "запись" и "связь". Тип "связь" определяется для двух типов "запись": предка и потомка. Переменные типа "связь" являются экземплярами связей.

Сетевая БД состоит из набора записей и набора соответствующих связей. На форматирование связи особых ограничений не накладывается. Если в иерархических структурах запись-потомок могла иметь только одну запись-предка, то в сетевой модели данных запись-потомок может иметь произвольное число записей-предков (свободных родителей).

Физическое размещение данных в базах сетевого типа может быть организовано практически теми же методами, что и в иерархических базах.

Представление связей в сетевой модели

К числу важнейших операций манипулирования данными баз сетевого типа можно отнести следующие:

- поиск записи в БД;
- переход от предка к первому потомку;
- переход от потомка к предку;
- создание новой записи;
- удаление текущей записи;
- обновление текущей записи;
- включение записи в связь;
- исключение записи из связи;
- изменение связей и т.д.

Достоинством сетевой модели данных является возможность эффективной реализации по показателям затрат памяти и оперативности. В сравнении с иерархической моделью сетевая представляет большие возможности в смысле допустимости образования произвольных связей.

Недостатком сетевой модели данных является высокая сложность и жесткость схемы БД, построенной на ее основе, а также сложность для понимания и выполнения обработки информации в БД обычным пользователем. Кроме того, в сетевой модели данных ослаблен контроль целостности связей вследствие допустимости установления произвольных связей между записями.

Системы на основе сетевой модели не получили широкого распространения на практике.

Постреляционная модель данных

Постреляционная модель данных представляет собой расширенную реляционную модель, снимающую ограничение неделимости данных, хранящихся в записях таблиц. Постреляционная модель данных допускает многозначные поля - поля, значения которых состоят из подзначений. Набор значений многозначных полей считается самостоятельной таблицей, встроенной в основную таблицу.

Помимо обеспечения вложенности полей постреляционная модель поддерживает ассоциированные многозначные поля (множественные группы). Совокупность ассоциированных полей называется ассоциацией. При этом в строке первое

значение одного столбца ассоциации соответствует первым значениям всех других столбцов ассоциации. Аналогичным образом связаны все вторые значения столбцов и т.д.

На длину полей и количество полей в записях таблицы не накладывается требование постоянства. Это означает, что структура данных и таблиц имеет большую гибкость.

Поскольку постреляционная модель допускает хранение в таблицах ненормализованных данных, возникает проблема обеспечения целостности и непротиворечивости данных. Эта проблема решается включением в СУБД механизмов, подобных хранимым процедурам в клиент-серверных системах.

Для описания функций контроля значений в полях имеется возможность создавать процедуры (коды конверсии и коды корреляции), автоматически вызываемые до или после обращения к данным. Коды корреляции выполняются сразу после чтения данных, перед их обработкой. Коды конверсии, наоборот, выполняются после обработки данных.

Достоинством постреляционной модели является возможность представления совокупности связанных реляционных таблиц одной постреляционной таблицей. Это обеспечивает высокую наглядность представления информации и повышение эффективности ее обработки.

Недостатком постреляционной модели является сложность решения проблемы обеспечения целостности и непротиворечивости хранимых данных.

Реляционные системы

Реляционные системы далеко не сразу получили широкое распространение. В то время как основные теоретические результаты в этой области были получены еще в 70-х годах и тогда же появились первые прототипы реляционных СУБД, долгое время считалось невозможным добиться эффективной реализации таких систем. Однако постепенное накопление методов и алгоритмов организации реляционных баз данных и управления ими привели к тому, что уже в середине 80-х годов реляционные системы практически вытеснили с мирового рынка ранние СУБД.

Реляционная модель данных основывается на математических принципах, вытекающих непосредственно из теории множеств и логики предикатов. Эти принципы впервые были применены в области моделирования данных в конце 1960-х гг. доктором *Е.Ф. Коддом*, в то время работавшим в IBM, а впервые опубликованы - в 1970 г. [1].

Техническая статья "Реляционная модель данных для больших разделяемых банков данных" доктора *Е.Ф. Кодда*, опубликованная в 1970 г., является родоначальницей современной теории реляционных БД. Доктор *Кодд* определил 12 правил реляционной модели (которые называют 12 правилами *Кодда*).

12 правил Кодда

1. Реляционная СУБД должна быть способна полностью управлять базой данных через ее реляционные возможности.
2. **Информационное правило** - вся информация в реляционной БД (включая имена таблиц и столбцов) должна определяться строго как значения в таблицах.
3. **Гарантированный доступ** - любое значение в реляционной БД должно быть гарантированно доступно для использования через комбинацию имени таблицы, значения первичного ключа и имени столбца
4. **Поддержка пустых значений** (null value) - СУБД должна уметь работать с пустыми значениями (неизвестными или неиспользованными значениями), в отличие от значений по умолчанию и независимо для любых *доменов*.
5. **Онлайновый реляционный каталог** - описание БД и ее содержания должны быть представлены на логическом уровне как таблицы, к которым можно применять запросы, используя язык базы данных.
6. **Исчерпывающий язык управления данными** - по крайней мере, один из поддерживаемых языков должен иметь четко определенный синтаксис и быть всеобъемлющим. Он должен поддерживать описание структуры данных и манипулирование ими, правила целостности, авторизацию и транзакции.
7. **Правило обновления представлений** (views) - все представления, теоретически обновляемые, могут быть обновлены через систему.
8. **Вставка, обновление и удаление** - СУБД поддерживает не только запрос на отбор данных, но и вставку, обновление и удаление
9. **Физическая независимость данных** - на программы-приложения и специальные программы логически не влияют изменения физических методов доступа к данным и структур хранилищ данных.
10. **Логическая независимость данных** - на программы-приложения и специальные программы логически не влияют, в пределах разумного, изменения структур таблиц.
11. **Независимость целостности** - язык БД должен быть способен определять правила целостности. Они должны сохраняться в онлайновом справочнике, и не должно существовать способа их обойти.
12. **Независимость распределения** - на программы-приложения и специальные программы логически не влияет, первый раз используются данные или повторно.
13. **Неподрывность** - невозможность обойти правила целостности, определенные через язык базы данных, использованием языков низкого уровня

Кодд предложил применение реляционной алгебры в СУРБД, для расчленения данных в связанные наборы. Он организовал свою систему БД вокруг концепции, основанной на наборах данных.

В реляционной модели данные разбиваются на наборы, которые составляют табличную структуру. Эта структура таблиц состоит из индивидуальных элементов данных, называемых полями. Одиночный набор или группа полей известна как запись.

Модель данных, или **концептуальное описание предметной области** - самый абстрактный уровень проектирования баз данных.

С точки зрения **теории реляционных БД**, основные принципы реляционной модели на концептуальном уровне можно сформулировать следующим образом:

- все данные представляются в виде упорядоченной структуры, определенной в виде строк и столбцов и называемой ***отношением*** ;
- все значения являются **скалярами**. Это означает, что для любой строки и столбца любого отношения существует одно и только одно значение;
- все операции выполняются над целым отношением, и результатом их выполнения также является целое отношение. Этот принцип называется **замыканием**

Формулируя принципы реляционной модели, доктор *Кодд* выбрал термин "отношение" (relation), потому что, по его мнению, этот термин однозначен (в то время как, например, термин "таблица" имеет множество различных видов - таблица в тексте, электронная таблица и пр.). Весьма распространено следующее заблуждение: *реляционная модель* названа так

потому, что она определяет связи между таблицами. На самом деле, название этой модели происходит от отношений (таблиц базы данных), лежащих в ее основе.

Каждая строка, содержащая данные, называется ***кортежем***, каждый столбец отношения называется ***атрибутом*** (на уровне практической работы с современными реляционными БД используются термины "запись" и "поле").

Элементами описания реляционной модели данных на концептуальном уровне
являются *сущности*, *атрибуты*, *домены* и связи

Сущность - некоторый обособленный объект или событие, информацию о котором необходимо сохранять в базе данных, имеющий определенный набор свойств - *атрибутов*. Сущности могут быть как физические (реально существующие объекты: например, СТУДЕНТ,*атрибуты* - № зачетной книжки, фамилия, его факультет, специальность, № группы и т.д.), так и абстрактные (например, ЭКЗАМЕН, *атрибуты* - дисциплина, дата, преподаватель, аудитория и пр.). Для *сущностей* различают ее тип и экземпляр. Тип характеризуется именем и списком свойств, а экземпляр - конкретными значениями свойств.

Атрибуты *сущности* бывают:

1. **Идентифицирующие и описательные**. Идентифицирующие *атрибуты* имеют уникальное значение для *сущностей* данного типа и являются *потенциальными ключами*. Они позволяют однозначно распознавать экземпляры *сущности*. Из *потенциальных ключей*выбирается один **первичный ключ** (ПК). В качестве ПК обычно выбирается потенциальный ключ, по которому чаще происходит обращение к *экземплярам записи*. ПК должен включать в свой состав минимально необходимое для идентификации количество *атрибутов*. Остальные ***атрибуты*** называются описательными.
2. **Простые и составные**. Простой *атрибут* состоит из одного компонента, его значение неделимо. Составной *атрибут* является комбинацией нескольких компонентов, возможно, принадлежащих разным типам данных (например, адрес). Решение о том, использовать составной*атрибут* или разбивать его на компоненты, зависит от особенностей процессов его использования и может быть связано с обеспечением высокой скорости работы с большими базами данных.
3. **Однозначные и многозначные** - могут иметь соответственно одно или много значений для каждого экземпляра сущности.
4. **Основные и производные**. Значение основного *атрибута* не зависит от других *атрибутов*. Значение производного *атрибута* вычисляется на основе значений других *атрибутов* (например, возраст человека вычисляется на основе даты его рождения и текущей даты

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

Домен - это набор всех допустимых значений, которые может содержать *атрибут*. Понятие ***"домен"*** часто путают с понятием "тип данных". Необходимо различать эти два понятия. Тип данных - это физическая концепция, а домен - логическая. Например, "целое число" - это тип данных, а "возраст" - это домен.

Связи - на концептуальном уровне представляют собой простые ассоциации между *сущностями*. Например, утверждение "Покупатели приобретают продукты" указывает, что между *сущностями* "Покупатели" и "Продукты" существует связь, и такие *сущности* называются**участниками** этой связи.

Существует несколько типов связей между двумя *сущностями*: это связи " **один к одному** ", " **один ко многим** " и " **многие ко многим** ".

Каждая связь в реляционной модели характеризуется именем, обязательностью, типом и степенью.
Различают **факультативные** и**обязательные** связи. Если *сущность* одного типа оказывается по необходимости связанной с *сущностью* другого типа, то между этими типами объектов существует обязательная связь (обозначается двойной линией). Иначе связь является факультативной.

Степень связи определяется количеством *сущностей*, которые охвачены данной связью. Пример *бинарной связи* - связь между отделом и сотрудниками, которые в нем работают.

Диаграмма "сущности-связи" (*Entity-Relationship diagrams*, или E/R diagram) служит для описания схемы базы на концептуальном уровне проектирования. Метод был предложен в 1976 г. Питером Пин Шань Ченом (Peter Pin Shan Chen) [2]. На диаграммах "сущности-связи" *сущности*изображаются в виде прямоугольников, *атрибуты* - в виде эллипсов, а связи - в виде ромбов (см. рис. 2.6).

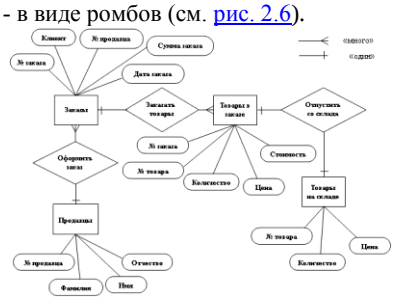


Рис. 2.6. Диаграмма "сущности-связи"

В дальнейшем многими авторами были разработаны свои варианты подобных моделей (нотация Мартина, нотация *IDEF1X*, нотация Баркера и др.). Кроме того, различные программные средства, реализующие одну и ту же нотацию, могут отличаться своими возможностями. По сути, все варианты диаграмм "сущность-связь" исходят из одной идеи - рисунок всегда нагляднее текстового описания. Все такие диаграммы используют графическое изображение *сущностей* предметной области, их свойств (*атрибутов*), и взаимосвязей между *сущностями*.

Проектирование схемы БД должно решать задачи минимизации дублирования данных, упрощения и ускорения процедур их обработки и обновления. При неправильно спроектированной схеме БД могут возникнуть аномалии модификации данных. Для решения подобных проблем проводится **нормализация отношений**

Однако в технологии работы с хранилищами данных может использоваться обратный прием - **денормализация отношений** с целью увеличения скорости выполнения запросов к очень большим объемам архивных данных.

В рамках реляционной модели данных *Э.Ф. Коддом* были разработаны принципы нормализации отношений и предложен механизм, позволяющий любое *отношение* преобразовать к *третьей нормальной форме*.

Нормализация - это формальный метод анализа *отношений* на основе их первичного ключа и существующих связей. Ее задача - это замена одной схемы (или совокупности *отношений*) БД другой схемой, в которой *отношения* имеют более простую и регулярную структуру.

При работе с реляционной моделью для создания *отношений* приемлемого качества достаточно выполнения требований первой нормальной формы.

Первая нормальная форма (1НФ) связана с понятиями простого и сложного *атрибутов*. Простой ***атрибут*** - это *атрибут*, значения которого атомарны (т.е. неделимы). Сложный *атрибут* может иметь значение, представляющее собой объединение нескольких значений одного или разных доменов. В первой нормальной форме устраняются повторяющиеся *атрибуты* или группы *атрибутов*, т.е. производится выявление неявных *сущностей*, "замаскированных" под *атрибуты*.

Отношение приведено к 1НФ, если все его атрибуты - простые, т.е. значение *атрибута* не должно быть множеством или повторяющейся группой.

Для приведения таблиц к *1НФ* необходимо разбить сложные *атрибуты* на простые, а многозначные *атрибуты* вынести в отдельные *отношения*.

Вторая нормальная форма (2НФ) применяется к *отношениям* с составными ключами (состоящими из двух и более *атрибутов*) и связана с понятиями функциональной зависимости.

Если в любой момент времени каждому значению *атрибута* A соответствует единственное значение *атрибута* B, то B функционально зависит от A (A ⟶ B). Атрибут (группа ***атрибутов***) A называется *детерминантом*.

Во второй нормальной форме устраняются *атрибуты*, зависящие только от части уникального ключа. Эта часть уникального ключа определяет отдельную *сущность*.

Отношение находится во 2НФ, если оно приведено к 1НФ и каждый неключевой атрибут функционально полно зависит от составного первичного ключа.

Третья нормальная форма (3НФ) связана с понятием транзитивной зависимости. Пусть A, B, C - *атрибуты* некоторого *отношения*. При этом A ⟶ B и B ⟶ C, но обратное соответствие отсутствует, т.е. C не зависит от B или B не зависит от A. Тогда говорят, что C транзитивно зависит от A (A ⟶ ⟶ C).

В *третьей нормальной форме* устраняются *атрибуты*, которые зависят от *атрибутов*, не входящих в уникальный ключ. Эти *атрибуты* являются основой отдельной *сущности*.

Отношение находится в 3НФ, если оно находится во 2НФ и не имеет атрибутов, не входящих в первичный ключ и находящихся в транзитивной зависимости от первичного ключа.

Существуют также *нормальная форма Бойса-Кодда* (НФБК), 4НФ и 5НФ. Однако наибольшее значение имеет *1НФ*, т.к. последующие НФ связаны с понятиями о составных ключах и сложных зависимостях от ключей, а на практике встречаются обычно более простые случаи.

Моделирование структуры базы данных при помощи алгоритма *нормализации* имеет серьезные недостатки:

1. Методика *нормализации* предполагает *первоначальное размещение* всех *атрибутов* проектируемой предметной области в одном *отношении*, что является очень неестественной операцией. Интуитивно разработчик сразу проектирует несколько *отношений* в соответствии с обнаруженными *сущностями*. Даже если совершить насилие над собой и создать одно или несколько *отношений*, включив в них все предполагаемые *атрибуты*, то совершенно неясен смысл полученного *отношения*.
2. Невозможно сразу определить полный список *атрибутов*. Пользователи имеют привычку называть разными именами одни и те же вещи или наоборот, называть одними именами разные вещи.
3. Для проведения процедуры *нормализации* необходимо выделить зависимости *атрибутов*, что тоже очень нелегко.

В реальном проектировании структуры базы данных применяются другой метод - так называемое **семантическое моделирование**. Семантическое моделирование представляет собой моделирование структуры данных, опирающееся на смысл этих данных. В качестве инструмента семантического моделирования используются различные варианты **диаграмм "сущность-связь (ERD)"** с построением концептуальной модели базы данных.

Любой специалист, освоивший общие принципы оптимальной организации реляционных баз данных, в состоянии построить модель, не противоречащую принципам *нормализации*.

Реляционная БД на физическом уровне состоит из таблиц, между которыми могут существовать связи по ключевым значениям. Одновременно с таблицами и информацией о связях в реляционной базе данных могут присутствовать "хранимые процедуры" и, в частности, "триггеры", обеспечивающие соблюдение условий ссылочной целостности базы.

Соблюдение условий ссылочной целостности в реляционной базе данных

Правило соответствия внешних ключей первичным - основное правило соблюдения условий ссылочной целостности. Для каждого значения внешнего ключа должно существовать соответствующее значение первичного ключа в родительской таблице

Ссылочная целостность может нарушиться в результате операций вставки (добавления), обновления и *удаления записей в таблицах*. В определении ссылочной целостности участвуют две таблицы - родительская и дочерняя, для каждой из них возможны эти операции, поэтому существует шесть различных вариантов, которые могут привести либо не привести к нарушению ссылочной целостности.

1. Обновление записей в родительской таблице.
2. Удаление записей в родительской таблице.
3. Вставка записей в дочерней таблице.
4. Обновление записей в дочерней таблице.

Постреляционные базы данных

В настоящее время известны также так называемые **"постреляционные"** СУБД, в основе которых лежат модель данных в виде многомерных таблиц (например в системе **Cache** фирмы InterSystems Corporation) и широкое использование принципов объектно-ориентированного подхода при организации баз данных и программировании.

Серверы баз данных

В локальных и глобальных компьютерных сетях широко применяются серверы: компьютеры и *программные средства* для обслуживания клиентов - рабочих станций и/или других серверов.

Примерами серверов могут быть:

- файловый сервер, поддерживающий общее хранилище файлов для всех рабочих станций;
- интернет-сервер, обеспечивающий предоставление информации в глобальной сети Интернет;
- почтовый сервер, обеспечивающий работу с электронной почтой;

- *сервер баз данных* - СУБД, которая принимает запросы по локальной сети и возвращает информацию, соответствующую запросу.

Термин " *сервер баз данных* " обычно используют для обозначения всей *СУБД*, основанной на архитектуре " *клиент-сервер* ", включая и серверную, и клиентскую части. Наиболее распространенными серверами являются в настоящее время Microsoft *SQL Server*, *Oracle*, *IBM DB2*Universal DataBase, Informix и др. Размер одной *базы данных* на этих серверах может достигать миллиона терабайт.

Раздел 3. Реляционные модели и языки. (комп. презентация – 4часа)

Реляционная алгебра. Основные и дополнительные операции реляционной алгебры. Реляционная модель данных. Целостность БД. Язык манипулирования данными для реляционной модели. Условия и ограничения, накладываемые на отношения реляционной моделью данных. Преимущества реляционной БД (в историческом аспекте).

Реляционная алгебра — это теоретический язык операций, позволяющих создавать на основе одного или нескольких отношений другое отношение без изменения самих исходных отношений. Таким образом, оба операнда и результат являются отношениями, поэтому результаты одной операции могут применяться в другой операции. Это позволяет создавать вложенные выражения реляционной алгебры (по аналогии с тем, как создаются вложенные арифметические выражения), но при любой глубине вложенности результатом является отношение. Такое свойство называется замкнутостью. Оно подчеркивает то, что применение любого количества операций реляционной алгебры к отношениям не приводит к созданию иных объектов, кроме отношений, точно так же, как результатами арифметических операций с числами являются только числа.

Реляционная алгебра является языком последовательного использования отношений, в котором все кортежи, возможно, даже взятые из разных отношений, обрабатываются одной командой, без организации циклов. Для команд реляционной алгебры предложено несколько вариантов синтаксиса. Ниже мы воспользуемся общепринятыми символическими обозначениями для этих команд и представим их в неформальном виде.

Основные и дополнительные операции реляционной алгебры

Существует несколько вариантов выбора операций, которые включаются в реляционную алгебру. Первоначально Кодд предложил восемь операций, но впоследствии к ним были добавлены и некоторые другие. Пять основных операций реляционной алгебры, а именно **выборка** *(selection)*, **проекция** *(projection)*, **декартово произведение** *(cartesian product)*,**объединение** *(union)* и **разность множеств** *(set difference)*, выполняют большинство действий по извлечению данных, которые могут представлять для нас интерес. На основании пяти основных операций можно также вынести дополнительные операции, такие как **операции соединения** *(join)*, **пересечения** *(intersection)* и **деления** *(division)*, которые могут быть выражены в терминах пяти основных операций.

Операции выборки и проекции являются ***унарными***, поскольку они работают с одним отношением. Другие операции работают с парами отношений, и поэтому их называют ***бинарными операциями***. В приведенных ниже определениях **R** и **S** — это два отношения, определенные на атрибутах $\mathrm{A} = (a_1, a_2, \ldots a_N)$ и $\mathrm{B} = (b_1, b_2, \ldots b_M)$ соответственно.

Операция	Обозначение	Область применения
Выборка	$\sigma_{предикат}(R)$	Определяет результирующее отношение, которое содержит только те кортежи (строки) из отношения R, которые удовлетворяют заданному условию (предикату)
Проекция	$\prod_{a_1,\ldots,a_n}(R$	Определяет новое отношение, содержащее вертикальное подмножество отношения R, создаваемое посредством извлечения значений указанных атрибутов и исключения из результата строк-дубликатов
Объединение	$R \cup S$	Определяет новое отношение, которое включает все кортежи, содержащиеся только в R, только в S, одновременно в R и S, причем все дубликаты кортежей исключены. При этом отношения R и S должны быть совместимыми по объединению
Разность	$R - S$	Разность двух отношений R и s состоит из кортежей, которые имеются в отношении R. но отсутствуют в отношении S. Причем отношения R и s должны быть совместимыми по объединению
Пересечение	$R \cap S$	Определяет отношение, которое содержит кортежи, присутствующие как в отношении R, так и в отношении S. Отношения R и S должны быть совместимыми по объединению
Декартово произведение	$R \times S$	Определяет новое отношение, которое является результатом конкатенации (т.е. сцепления) каждого кортежа из отношения R с каждым кортежем из отношения S
Тета-соединение	$R \bowtie_F S$	Определяет отношение, которое содержит кортежи из декартова произведения отношений R и S, удовлетворяющие предикату F
Соединение по эквивалентности	$R \bowtie_F S$	Определяет отношение, которое содержит кортежи из декартова произведения отношений R и S, удовлетворяющие предикату F (предикат должен предусматривать только сравнение на равенство)
Естественное соединение	$R \bowtie S$	Естественным соединением называется соединение по эквивалентности двух отношений R и S, выполненное по всем общим атрибутам x, из результатов которого исключается по одному экземпляру каждого общего атрибута
(Левое) внешнее соединение	$R \supset\!\lhd S$	Соединение, при котором кортежи отношения R, не имеющие совпадающих значений в общих столбцах отношения S, также включаются в результирующее отношение
Полусоединение	$R \rhd_F S$	Определяет отношение, содержащее те кортежи отношения R, которые входят в соединение отношений R и S
Деление	$R \div S$	Определяет отношение, состоящее из множества кортежей отношения R, которые

		определены на атрибуте C, соответствующем комбинации всех кортежей отношения S, где C — множество атрибутов, имеющихся в отношении R, но отсутствующих в отношении S

Реляционная модель данных

Реляционная модель впервые была предложена Э. Ф. Коддом (E. F. Codd) в 1970 году в его основополагающей статье "Реляционная модель данных для больших совместно используемых банков данных". В настоящее время публикацию этой статьи принято считать поворотным пунктом в истории развития систем баз данных, хотя следует заметить, что еще раньше была предложена модель, основанная на множествах. Цели создания реляционной модели формулировались следующим образом:

- Обеспечение более высокой степени независимости от данных. Прикладные программы не должны зависеть от изменений внутреннего представления данных, в частности от изменений организации файлов, переупорядочивания записей и путей доступа.
- Создание прочного фундамента для решения семантических вопросов, а также проблем непротиворечивости и избыточности данных. В частности, в статье Кодда вводится понятие нормализованных отношений, т.е. отношений без повторяющихся групп. (Процесс нормализации обсуждается в ниже)
- Расширение языков управления данными за счет включения операций над множествами.

Кроме того, позже были предложены некоторые расширения реляционной модели данных, предназначенные для наиболее полного и точного выражения смысла данных, для поддержки объектно-ориентированных понятий, а также для поддержки дедуктивных возможностей.

Терминология, используемая » реляционной модели, порой может привести к путанице, поскольку помимо предложенных двух наборов терминов существует еще один — третий. Отношение в нем называется файлом (file), кортежи — записями (records), а атрибуты — полями (fields). Эта терминология основана на том факте, что физически реляционная СУБД может хранить каждое отношение в отдельном файле. В таблице показаны соответствия, существующие между тремя упомянутыми выше группами терминов.

Официальные термины	Альтернативный вариант 1	Альтернативный вариант 2
Отношение	Таблица	Файл
Кортеж	Строка	Запись
Атрибут	Столбец	Поле

Целостность БД

Реляционная модель данных имеет две части: управляющую часть, которая определяет типы допустимых операций с данными, и набор ограничений целостности, которые гарантируют корректность данных. Поскольку каждый атрибут связан с некоторым доменом, для множества допустимых значений каждого атрибута отношения определяются так называемые ограничения домена. Помимо этого, задаются два важных правила целостности, которые, по сути, являются ограничениями для всех допустимых состояний базы данных. Эти два основных правила реляционной модели называются целостностью сущностей и ссылочной целостностью.

Целостность сущностей

Первое ограничение целостности касается первичных ключей базовых отношений. Здесь базовое отношение определяется как отношение, которое соответствует некоторой сущности в концептуальной схеме.

Целостность сущностей: В базовом отношении ни один атрибут первичного , ключа не может содержать отсутствующих значений, обозначаемых как NULL.

По определению, первичный ключ — это минимальный идентификатор, который используется для уникальной идентификации кортежей. Это значит, что никакое подмножество первичного ключа не может быть достаточным для уникальной идентификации кортежей. Если допустить присутствие NULL в любой части первичного ключа, это равносильно утверждению, что не все его атрибуты необходимы для уникальной идентификации кортежей, что противоречит определению первичного ключа.

Ссылочная целостность

Второе ограничение целостности касается внешних ключей.

Ссылочная целостность: Если в отношении существует внешний ключ, то значение внешнего ключа должно либо соответствовать значению потенциального ключа некоторого кортежа в его базовом отношении либо внешний ключ должен полностью состоять из значений NULL.

Например, считается допустимым создание записи с информацией о новом сотруднике с указанием NULL вместо номера отделения, в котором этот сотрудник работает. Такая ситуация может иметь место в том случае, когда сотрудник зачислен в штат компании, но еще не приписан к какому-то конкретному отделению.

Корпоративные ограничения целостности

Корпоративные ограничения целостности: Дополнительные правила поддержки целостности данных, определяемые пользователями или администраторами базы данных.

Пользователи сами могут указывать дополнительные ограничения, которым должны удовлетворять данные. Например, если в одном отделении не может работать больше 20 сотрудников, то пользователь может указать это как правило, а СУБД должна следить за его выполнением. К сожалению, уровень поддержки реляционной целостности в разных системах существенно изменяется.

Преимущества и недостатки СУБД

СУБД обладают как многообещающими потенциальными преимуществами, так и недостатками, которые мы кратко рассмотрим в этом разделе.

Преимущества

Преимущества систем управления базами данных перечислены в таблице:

Преимущество
Контроль за избыточностью данных

Непротиворечивость данных
Больше полезной информации при том же объеме хранимых данных
Совместное использование данных
Поддержка целостности данных
Повышенная безопасность
Применение стандартов
Повышение эффективности с ростом масштабов системы
Возможность нахождения компромисса при противоречивых требованиях
Повышение доступности данных и их готовности к работе
Улучшение показателей производительности
Упрощение сопровождения системы за счет независимости отданных
Улучшенное управление параллельной работой
Развитые службы резервного копирования и восстановления

Недостатки

Недостатки подхода, связанного с применением баз данных, перечислены в таблице: Сложность Размер Стоимость СУБД Затраты на преобразование Производительность

Раздел 4. Предметная область базы данных и ее модели.

Аннотация: В настоящей лекции вводится понятие предметной области базы данных, описываются основные приемы построения моделей предметной области. Рассматриваемые модели являются входными данными для процесса проектирования базы данных.

Понятие предметной области

Основным назначением информационных систем является оперативное обеспечение пользователя информацией о внешнем мире путем реализации вопросно-ответного отношения. Вопросно-ответные отношения, получая интерпретацию во внешнем мире (мире вне информационной системы), позволяют выделить для информационной системы определенный его фрагмент - предметную область, - который будет воплощен в *автоматизированной информационной системе*. *Информация* о внешнем мире представляется в информационной системе (ИС) в форме данных. Это ограничивает возможности смысловой интерпретации информации и конкретизирует семантику ее представления в ИС. Совокупность этих выделенных для ИС данных, связей между ними и операций над ними образует *информационную* и *функциональную*

Понятие *предметной области базы данных* является одним из базовых понятий информатики и не имеет точного определения. Его использование в контексте ИС предполагает существование устойчивой во времени соотнесенности между именами, понятиями и определенными реалиями внешнего мира, не зависящей от самой ИС и ее круга пользователей. Таким образом, введение в рассмотрение понятия *предметной области базы данных* ограничивает и делает обозримым *пространство* информационного поиска в ИС и позволяет выполнять запросы за конечное время.

Совокупность реалий (объектов) внешнего мира - объектов, о которых можно задавать вопросы, - образует объектное *ядро предметной области*, которое имеет онтологический статус. Нельзя получить в ИС ответ на вопрос о том, что ей неизвестно. Термин *объект* является первичным, неопределяемым понятием. Синонимами термина "*объект*" являются "реалия, сущность, вещь". Однако термин сущность понимается нами несколько уже, как *компонент модели предметной области*, т.е. как уже выделенный на концептуальном уровне *объект* для*базы данных*. Таким образом, выделяемые в *предметной области* объекты превращаются аналитиками (а не проектировщиками *базы данных*) в сущности. *Сущность предметной области* является результатом абстрагирования реального объекта путем выделения и фиксации набора его свойств. Сущность является результатом абстрагирования реального объекта, т.е. в нашем контексте имеет гносеологический статус. Хотя далее в контексте сущность нередко отождествляется с объектом.

На рис. 2.1 представлен один из подходов к классификации объектов *предметной области*.

Рис. 2.1. Классификации объектов предметной области

Примерами сущностей (с точки зрения ИС) или объектов (с точки зрения внешнего мира) являются отдельный студент, *группа* студентов, аудитория, время занятий, слова, числа, символы. Обычно считается, что быть объектом - это значит быть дискретным и различимым. Примеры "не-объектов" - это мир, время, смысл, хотя и такие категории могут сохраняться в базе данных.

Объекты взаимодействуют между собой через свои свойства, что порождает ситуации. Ситуации - это взаимосвязи, выражающие взаимоотношения между объектами. Ситуации в *предметной области* описываются посредством высказываний о *предметной области* с использованием исчисления высказываний и *исчисления предикатов*, т.е. формальной, математической логики. Например, *высказывание*"Программист и *менеджер* есть служащие компании" описывает *отношение* включения. Таким образом, вся *информация* об объектах и сущностях *предметной области* описывается с помощью утверждений на естественном языке.

Отметим, что в семантике естественных языков ситуация и взаимосвязь считаются почти синонимами. Ситуация содержит *высказывание* об объектах *предметной области*, которому можно приписать некоторую оценку истинности и представить в виде предиката после введения переменных. Таким образом, совокупность высказываний о *предметной области* можно трактовать как *определение* информационного пространства для *базы данных*.

На рис. 2.2 представлен один из подходов к классификации ситуаций в рамках *предметной области*.

Рис. 2.2. Классификация ситуаций предметной области

Различают статические и динамические ситуации. Примерами статических ситуаций являются такие ситуации, как иметь цвет, иметь возраст. Примерами динамических ситуаций являются такие ситуации, как создать утюг, выпечь хлеб.

Обратите внимание на то, что ситуация также может представлять собой *объект* (см. рис. 2.1) и обладать свойствами. С другой стороны, приведенная классификация рассматривает свойства как специальный случай ситуаций. Подобная *коллизия* порождает неоднозначность при моделировании *предметной области базы данных*. Поставим вопрос - что есть цвет автомобиля? *Объект*, свойство, ситуация? К обсуждению этого вопроса мы вернемся специально в следующей лекции.

Приведенная классификация вводит в предметную область два важных аспекта - *пространство* и время, причем время и как момент, и как*интервал*. Предметная область существует в пространстве и во времени, т.е. ей присущи, как и реальному миру, временные и пространственные отношения и связи. Следует отличать *реальное время* внешнего мира и его отражение в базе данных и в источниках информации. В базе данных взаимосвязи, зависящие от времени, фиксируются только после их регистрации в базе данных. Таким образом, предметная область в каждый конкретный момент времени представляет собой выделенную совокупность определенных объектов и ситуаций, называемую *состоянием предметной области* (или снимком).

Введем *определение предметной области*.

Определение. Предметная область - это целенаправленная первичная трансформация картины внешнего мира в некоторую умозрительную картину, определенная часть которой фиксируется в ИС в качестве *алгоритмической модели* фрагмента действительности.

Понятие *предметной области* было введено в начале 80-х годов прошлого века, когда учеными в области ИС была осознана необходимость использовать *семантические модели* для представления информации в компьютерных системах. Так же как требования к компьютерной системе формируются средствами естественного языка, так и *информация* в компьютерных системах представляется средствами особого языка с определенной семантикой. Такой подход впервые был представлен П. Ченом в 1976 году.

Информационная модель предметной области базы данных

Одним из ключевых моментов создания ИС с целью автоматизации информационных процессов организации является всестороннее изучение объектов автоматизации, их свойств, взаимоотношений между этими объектами и *представление* полученной информации в виде информационной модели данных.

Информационная *модель данных* предназначена для представления семантики *предметной области* в терминах субъективных средств описания - сущностей, атрибутов, *идентификаторов сущностей*, *супертипов*, *подтипов* и т.д.

Информационная модель предметной области базы данных содержит следующие основные конструкции:

- диаграммы "*сущность-связь*" (Entity - *Relationship* Diagrams);
- определения сущностей;
- уникальные *идентификаторы сущностей* ;
- определения *атрибутов сущностей* ;
- отношения между сущностями;
- *супертипы* и подтипы.

Внимание! Элементы информационной модели данных предметной области являются входными данными для решения задачи*проектирования базы данных* - создания *логической модели данных*.

Сущности, атрибуты и идентификаторы (ключи) сущности, домены атрибутов

Предметом информационной модели является абстрагирование объектов или явлений реального мира в рамках предметной области, в результате которого выявляются сущности (entity) предметной области. Как правило, они обозначаются именем существительным естественного языка.

Сущность описывается с помощью данных, именуемых свойствами или *атрибутами (attributes) сущности*. Как правило, атрибуты являются определениями в высказывании о сущности и обозначаются именами существительными естественного языка. Сущности вступают в связи друг с другом через свои атрибуты. Каждая группа атрибутов, описывающих одно реальное проявление сущности, представляет собой экземпляр (instance) сущности. Иными словами, *экземпляры сущности* - это реализации сущности, отличающиеся друг от друга и допускающие однозначную идентификацию.

Внимание! При представлении сущности в базе данных хранятся только ее атрибуты.

Одним из основных компьютерных способов распознавания сущностей в базе данных является присвоение сущностям идентификаторов (Entity identifier). Часто *идентификатор сущности* называют ключом. Задача выбора *идентификатора сущности* является семантически субъективной задачей. Поскольку сущность определяется набором своих атрибутов, то для каждой сущности целесообразно выделить такое подмножество атрибутов, которое однозначно идентифицирует данную сущность.

Некоторые сущности имеют естественные идентификаторы. Например, естественным идентификатором *счета-фактуры* является его номер.*Идентификаторы сущности* могут быть составными - составленными из нескольких атрибутов и атомарными - составленными из одного атрибута сущности.

Идентификация сущностей проводится аналитиками. Однако чаще всего их решение не является окончательным! Задача проектировщика баз данных - обеспечить при сохранении *экземпляров сущности* в базе данных наличие у каждого ее нового экземпляра уникального идентификатора. *Уникальный идентификатор сущности* - это *атрибут сущности*, позволяющий отличать одну сущность от другой. Если сущность имеет несколько уникальных идентификаторов, так называемых *возможных ключей*, то проектировщик должен выбрать первичный ключ сущности.

Различают однозначные и многозначные атрибуты. Однозначными являются атрибуты, которые в пределах конкретного экземпляра сущности имеют только одно значение. В противном случае они считаются многозначными.

На уровне информационного моделирования данных назначение домена атрибуту носит общий характер. Например, атрибут текстовый, числовой, бинарный, дата или "не определен". В последнем случае аналитик должен дать описание домена. На последующих стадиях тип домена конкретизируется, смысл понятия домена в логической и физической моделях базы данных уже, чем его может понимать аналитик. Это связано с тем, что в рамках физической модели базы данных домен реализуется посредством механизма ограничения домена, СУБД не понимает неопределенных доменов.

Отношения, связи

Сущности не существуют отдельно друг от друга. Между ними имеются реальные отношения (*Relationship*), и они должны быть отражены в*информационной модели предметной области*. При выделении отношений акцент делается на фиксацию связей и их характеристик. Отношение (связь) представляет собой соединение (взаимоотношение) между двумя или более сущностями. Каждая связь реализуется через значения*атрибутов сущностей*. Обычно связь обозначается глаголом. Каждая связь также должна иметь свой уникальный *идентификатор связи*.

Внимание! В реляционной базе данных отношения реализуются только через ограничение целостности по внешнему ключу. Поэтому проектировщик базы данных должен проконтролировать, чтобы связь между сущностями осуществлялась через точно указанные атрибуты, которые будут определять *уникальный ключ* связи. Выбор ключей сущностей - одно из важнейших проектных решений, которое предстоит сделать проектировщику при переходе от *информационной модели предметной области* к логической модели базы данных.

Связи характеризуются *степенью связи* и классом принадлежности сущности к связи. Степень (*мощность) связи* - это отношение числа сущностей, участвующих в образовании связи. Например, "один-к-одному", "один-ко-многим", "многие-ко-многим". На уровне информационной модели допускается неопределенная или неразрешенная связь. *Класс принадлежности сущности* - это характер участия сущности в связи. Различают обязательные и необязательные *классы принадлежности сущности* к связи. Обязательным является такой класс принадлежности, когда *экземпляры сущности* участвуют в установлении связи в обязательном порядке. В противном случае сущность принадлежит к необязательному классу принадлежности. Для необязательного класса принадлежности сущности степень связи может быть равна нулю, т.е.*экземпляр сущности* можно связать с 0, 1 или несколькими экземплярами другой сущности. Для обязательного класса принадлежности степень связи не может равняться нулю.

Отношения, *связывающие сущность* саму с собой, называются *рефлексивными*. Типичным примером *рефлексивных* отношений является определение структуры подчиненности в отношении "Сотрудники". *Рефлексивные* отношения чаще всего отражают *иерархические отношения* внутри структуры данных. Они порождают ряд проблем проектирования, о которых речь пойдет позже.

С точки зрения *отношений различают* слабые (*weak*) сущности. Слабые сущности - это сущности, которые не могут присутствовать в базе данных, пока не существует связанного с ней экземпляра другой сущности. Примером такой сущности является заказ, который не может существовать без клиента. Слабые сущности имеют обязательный класс принадлежности, и степень связи такой сущности не может равняться нулю. Связь "заказ-клиент" является обязательной.

Выявление слабых сущностей и связанных с ними обязательных отношений необходимо для обеспечения целостности и согласованности данных. Так, например, неизвестному клиенту невозможно приписать заказ.

Подтипы и супертипы

Иногда выделенная сущность несет в себе отношение включения или часть-целое. При этом существует некоторый атрибут, значения которого порождают *разбиение множества экземпляров сущности* на непересекающиеся подмножества - категории сущности. Категории сущности называются *подтипами* и выделяют в подчиненную в рамках отношения сущность, которая является категорией исходной сущности.

Иногда из исходной сущности выделяются общие для полученных категорий атрибуты, и таким образом выделяется сущность, которая становится *супертипом*. За выделенной сущностью-супертипом обычно оставляют наименование исходной сущности, хотя ее семантический смысл меняется.

Супертип с порожденными им *подтипами* является примером так называемой составной сущности. Составная сущность является логической конструкцией модели для представления набора сущностей и связей между ними как единого целого.

Пример. Сущность Автомобиль можно разбить на следующие подтипы: автомобили с приводом на два колеса, автомобили с приводом на четыре колеса, автомобили с переключаемым приводом.

Для проектировщика базы данных важно знать, что все *экземпляры сущности- супертипа* относятся только к одному из ее подтипов. Наличие в модели подтипов и *супертипов* усложняют проектирование и создают определенные трудности в реализации.

Поэтому важно на ранней стадии проектирования установить, является ли наличие *супертипов* в модели необходимым. Для этого можно предпринять следующие действия:

- установить, много ли одинаковых свойств имеют различные подтипы. Следует помнить, что чем меньше подтипы похожи друг на друга, тем больше вероятность введения *супертипа*, или
- найти *экземпляр сущности*, который можно обоснованно включить в более чем один подтип. Поскольку это противоречит определению супертипа, то предлагаемое разбиение недопустимо.

Диаграммы "сущность-связь"

Типичной формой документирования *информационной модели предметной области* являются *диаграммы "сущность-связь"* (ER -диаграммы).*ER-диаграмма* позволяет графически представить все элементы информационной модели согласно простым, интуитивно понятным, но строго определенным правилам - нотациям. Далее мы будем пользоваться условными обозначениями, принятыми в методологии информационного проектирования.

Построение ER -диаграмм, как правило, ведется с использованием CASE -средств. Выбор CASE -средств и способы работы с ними в настоящем курсе не обсуждаются.

Документирование сущностей и атрибутов

Сущность на ER-диаграмме представляется прямоугольником с именем в верхней части. Будем использовать английские слова для именования элементов модели.

В прямоугольнике перечисляются *атрибуты сущности*, при этом атрибуты, составляющие *уникальный идентификатор сущности*, подчеркиваются.

Изучив и проверив качество информационной модели данных предметной области, представленной в виде набора ER-диаграмм, проектировщик базы данных может приступать к *созданию логической модели базы данных*.

Функциональная модель предметной области базы данных

Понятие функциональной модели предметной области базы данных

Вторым ключевым моментом создания ИС с целью автоматизации информационных процессов организации является анализ функционального взаимодействия объектов автоматизации. Результаты такого анализа многогранны. Аналитики представляют их в виде *функциональной моделипредметной области базы данных***.** *Функциональная модель предметной области* является собирательным понятием. Состав *функциональной модели* существенно зависит от контекста конкретного ИТ-проекта и может быть представлен посредством довольно широкого спектра документов в виде текстовой и графической информации. К рассмотрению таких документов проектировщик баз данных должен подходить с учетом следующих двух положений:

- главное назначение ИС является базовым критерием оценки достаточности предоставляемой информации;
- *функциональная модель* предназначена для описания *процессов обработки данных* в рамках выделенной предметной области с различных точек зрения.

Описать процессы обработки информации всесторонне с помощью одной *описательной модели* сложно. В последнее время предпринимаются некоторые успешные попытки разработать унифицированную модель в рамках *объектно-ориентированного анализа и проектирования* (*OOA*&D) с помощью UML-конструкций. В настоящих лекциях мы основываемся только на результатах структурного *анализа предметной области* как наиболее прагматичном подходе для проектирования классических реляционных баз данных. Использование *объектно-ориентированного подхода* к *проектированию баз данных* - это предмет отдельного курса лекций.

Определим *функциональную модель предметной области* базы данных как совокупность некоторых моделей, предназначенных для описания процессов обработки информации. Будем называть эти модели конструкциями *функциональной модели*. Ниже приведен перечень основных конструкций *функциональной модели*, необходимые для выполнения *проектирования реляционных баз данных***.**

- Модели процессов:
  - *бизнес-модель* процессов (*иерархия функций* системы);
  - модель *потока данных***.**
- Модели состояний:
  - *модель жизненного цикла сущности* ;
  - набор спецификаций функций системы (требования);
  - описание функций системы через сущности и атрибуты;
  - бизнес-правила, которые реализуют функции.

Внимание! Элементы *информационной модели предметной области* являются входными данными для задачи создания *логической модели данных*. Элементы *функциональной модели предметной области* являются входными данными для задачи проектирования приложений базы данных и, частично, для задачи *создания физической модели базы данных***.**

Диаграмма потока данных позволяет:

- представить систему с точки зрения источников и потребителей данных;
- показать перемещение данных в процессе их обработки;
- показать внешние механизмы подачи данных;
- показать метод сбора данных.

Диаграмма потока данных предоставляет проектировщику баз данных информацию:

- *хранилищах данных*, что позволит на последующих стадиях проектирования обоснованно определить число баз данных для информационной системы;
- принятых схемах преобразования информации в процессе ее обработки, что позволит в ходе проектирования приложений составить спецификацию приложений. Спецификации приложений вместе с диаграммами потоков данных передаются разработчикам приложений.

Раздел 5. Жизненный цикл БД. (компьютерная презентация – 2 час)

Жизненные циклы информационных систем. Цели и задачи проектирования. Проектирование баз данных (о трех этапах). Формулирование и анализ требований. Концептуальное проектирование. Модель «сущность-связь». Критерии выбора первичного ключа.

Жизненные циклы информационных систем

Начиная с 1970-х годов системы баз данных стали постепенно заменять файловые системы, использовавшиеся как часть инфраструктуры информационных систем (Information System — IS) организаций. Параллельно с этим росло признание того факта, что данные являются важным корпоративным ресурсом, к которому нужно относиться так же бережно, как и к другим ресурсам организации. Это привело к тому, что во многих организациях появились целые отделы или функциональные подразделения, занимавшиеся администрированием данных (АД) и администрированием баз данных (АБД). Они отвечали за обработку и управление корпоративными данными и корпоративными базами данных.

База данных является фундаментальным компонентом информационной системы, а ее разработку и использование следует рассматривать с точки зрения самых широких требований организации. Следовательно, жизненный цикл информационной системы организации неотъемлемым образом связан с жизненным циклом системы базы данных, поддерживающей ее функционирование. Жизненный цикл информационной системы обычно состоит из нескольких этапов: планирование, сбор и анализ требований, проектирование, создание прототипа, реализация, тестирование, преобразование данных и сопровождение.

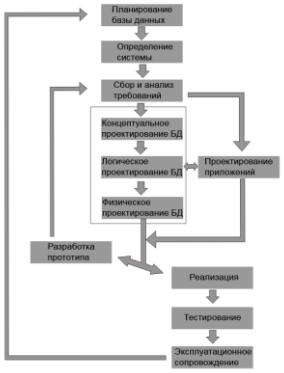
Все этапы жизненного цикла информационной системы здесь рассматриваются с точки зрения разработки приложения баз данных. Однако следует отметить, что разработку любого приложения базы данных всегда полезно рассматривать с более широкой точки зрения — как разработку определенного компонента всей информационной системы организации в целом.

Жизненный цикл приложения баз данных

Как уже упоминалось выше, система базы данных является фундаментальным компонентом более широкого понятия — информационной системы организации. Следовательно, жизненный цикл приложений баз данных неразрывно связан с жизненным циклом информационной системы. Этапы жизненного цикла приложения базы данных показаны на рисунке. Следует признать, что эти этапы не являются строго последовательными, а предусматривают в некоторых случаях возврат к предыдущим этапам с помощью обратных связей (feedback loops). Например, при проектировании базы данных могут

возникнуть проблемы, для разрешения которых потребуется вернуться к этапу сбора и анализа требований. Обратные связи могут возникать почти между всеми этапами, но на рисунке показаны только наиболее важные из них. Основные сведения о наиболее важных мероприятиях, связанных с реализацией каждого этапа жизненного цикла приложения базы данных, приведены в таблице.

жизненного цикла приложения базы данных.



Основные действия, выполняемые на каждом этапе жизненного цикла приложения базы данных:

Этап	Описание
Планирование разработки базы данных	Планирование наиболее эффективного способа реализации этапов жизненного цикла системы
Определение требований к системе	Определение диапазона действий и границ приложения базы данных, состава его пользователей и областей применения
Сбор и анализ требований пользователей	Сбор и анализ требований пользователей из всех возможных областей применения
Проектирование базы данных	Полный цикл разработки включает концептуальное, логическое и физическое проектирование базы данных
Выбор целевой СУБД (необязательный этап)	Выбор наиболее подходящей СУБД для приложения базы данных
Разработка приложений	Определение пользовательского интерфейса и прикладных программ, которые используют и обрабатывают данные в базе данных
Создание прототипов (необязательный этап)	Создание рабочей модели приложения базы данных, которая позволяет разработчикам или пользователям представить и оценить окончательный вид и способы функционирования системы
Реализация	Создание внешнего, концептуального и внутреннего определений базы данных и прикладных программ
Преобразование и загрузка данных	Преобразование и загрузка данных (и прикладных программ) из старой системы в новую
Тестирование	Приложение базы данных тестируется с целью обнаружения ошибок, а также его проверки на соответствие всем требованиям, выдвинутым пользователями
Эксплуатация и сопровождение	На этом этапе приложение базы данных считается полностью разработанным и реализованным. Впредь вся система будет находиться под постоянным наблюдением и соответствующим образом поддерживаться, В случае необходимости в функционирующее приложение могут вноситься изменения, отвечающие новым требованиям. Реализация этих изменений проводится посредством повторного выполнения некоторых из перечисленных выше этапов жизненного цикла

Цели и задачи проектирования

В настоящее время ключевая роль в достижении успеха большинства компьютеризованных систем принадлежит не используемому оборудованию, а программному обеспечению. Однако существующие исторические свидетельства о разработке программного обеспечения систем не производят столь глубокого впечатления, как хронологические обзоры стремительного прогресса в области аппаратных средств вычислительной техники. В последние десятилетия прикладные программы проделали путь от маленьких и сравнительно простых приложений из нескольких строк кода до очень больших и сложных приложений, состоящих из

Подходы к проектированию базы данных

Существуют два основных подхода к проектированию систем баз данных: нисходящий и восходящий. При восходящем подходе работа начинается с самого нижнего уровня атрибутов (т.е. свойств сущностей и связей), которые на основе анализа существующих между ними связей группируются в отношения, представляющие типы сущностей и связи между ними. Например процесс нормализации представляет собой вариант восходящего подхода при проектировании баз данных. Нормализация предусматривает идентификацию требуемых атрибутов с последующим созданием из них нормализованных таблиц, основанных на функциональных зависимостях между этими атрибутами.

Восходящий подход в наибольшей степени приемлем для проектирования простых баз данных с относительно небольшим количеством атрибутов. Однако использование этого подхода существенно усложняется при проектировании баз данных с большим количеством атрибутов, установить среди которых все существующие функциональные зависимости довольно затруднительно. Поскольку концептуальная и логическая модели данных для сложных баз данных могут содержать от сотен до тысяч атрибутов, очень важно выбрать подход, который помог бы упростить этап проектирования.

Кроме того, на начальных стадиях формулирования требований к данным в крупной базе данных может быть трудно установить все атрибуты, которые должны быть включены в модели данных.

Более подходящей стратегией проектирования сложных баз данных является использование нисходящего подхода. Начинается этот подход с разработки моделей данных, которые содержат несколько высокоуровневых сущностей и связей, затем работа продолжается в виде серии нисходящих уточнений низкоуровневых сущностей, связей и относящихся к ним атрибутов. Нисходящий подход демонстрируется в концепции модели "сущность-связь". В этом случае работа начинается с выявления сущностей и связей между ними, интересующих данную организацию в наибольшей степени. Например, сначала можно было бы идентифицировать сущности PrivateQwner (Владелец) и PropertyForRent (Объект недвижимости), затем установить между ними связь PrivateOwner Owns (Владеет) PropertyForRent и лишь после этого определить связанные с ними атрибуты — например, PrivateOwner {ownerNo, name, address) и PropertyForRent (propertyNo, address).

Кроме этих подходов для проектирования баз данных могут применяться другие подходы, например, подход "от общего к частному" или "смешанная стратегия проектирования". Подход "от общего к частному" напоминает восходящий подход, но отличается от него тем, что вначале выявляется набор основных сущностей с последующим расширением круга рассматриваемых сущностей, связей и атрибутов, которые взаимодействуют с первоначально определенными сущностями. В смешанной стратегии сначала используются восходящий и нисходящий подходы для создания разных частей модели, после чего все подготовленные фрагменты собираются в единое целое.

Моделирование данных

Основные цели моделирования данных состоят в изучении значения (семантики) данных и упрощении процедур описания требований к данным. При создании модели данных необходимо получить ответы на определенные вопросы об отдельных сущностях, связях и атрибутах. Полученные дополнительные сведения помогут разработчикам раскрыть особенности семантики корпоративных данных, которые существуют независимо от того, отмечены они в формальной модели данных или нет. Сущности, связи и атрибуты являются фундаментальными информационными объектами любого предприятия. Однако их реальный смысл будет оставаться не вполне понятным до тех пор, пока они не будут должным образом описаны в документации. Моделирование данных упрощает понимание смысла элементов данных, поэтому создание модели необходимо для того, чтобы гарантировать понимание следующих аспектов данных:

- требования к данным отдельных пользователей;
- характер самих данных независимо от их физического представления;
- использование данных в пределах области применения приложения.

Критерии оценки модели данных

Оптимальная модель данных должна удовлетворять критериям, перечисленным в таблице. Однако иногда эти критерии несовместимы, поэтому приходится идти на некоторый компромисс. Например, в погоне за наибольшей выразительностью модели данных можно утратить ее простоту.

Критерий	Описание
Структурная достоверность	Соответствие способу определения и организации информации на данном предприятии
Простота	Удобство изучения модели как профессионалами в области разработки информационных систем, так и обычными пользователями
Выразительность	Способность представлять различия между данными, связи между данными и ограничения
Отсутствие избыточности	Исключение излишней информации, т.е. любая часть данных должна быть представлена только один раз
Способность к совместному использованию	Отсутствие принадлежности к какому-то особому приложению или технологии и, следовательно, возможность использования модели во многих приложениях и технологиях
Расширяемость	Способность развиваться и включать новые требования с минимальным воздействием на работу уже существующих приложений
Целостность	Согласованность со способом использования и управления информацией внутри предприятия
Схематическое представление	Возможность представления модели с помощью наглядных схематических обозначений

Этапы проектирования базы данных

Процесс проектирования базы данных состоит из трех основных этапов: ***концептуальное, логическое и физическое проектирование.***

Концептуальное проектирование базы данных

Концептуальное проектирование базы данных. Процесс создания модели используемой на предприятии информации, не зависящей от любых физических аспектов ее представления.

Первый этап процесса проектирования базы данных называется концептуальным проектированием базы данных. Он заключается в создании концептуальной модели данных для анализируемой части предприятия. Эта модель данных создается на основе информации, записанной в спецификациях требований пользователей. Концептуальное проектирование базы данных абсолютно не зависит от таких подробностей ее реализации, как тип выбранной целевой СУБД, набор создаваемых прикладных программ, используемые языки программирования, тип выбранной вычислительной платформы, а также от любых других особенностей физической реализации.

При разработке концептуальная модель данных постоянно подвергается тестированию и проверке на соответствие требованиям пользователей. Созданная концептуальная модель данных предприятия является источником информации для этапа логического проектирования базы данных.

Логическое проектирование базы данных

Логическое проектирование базы данных. Процесс создания модели используемой на предприятии информации на основе выбранной модели организации данных, но без учета типа целевой СУБД и других физических аспектов реализации.

Второй этап проектирования базы данных называется логическим проектированием базы данных. Его цель состоит в создании логической модели данных для исследуемой части предприятия. Концептуальная модель данных, созданная на предыдущем этапе, уточняется и преобразуется в логическую модель данных.Логическая модель данных учитывает особенности выбранной модели организации данных в целевой СУБД (например, реляционная модель).

Если концептуальная модель данных не зависит от любых физических аспектов реализации, то логическая модель данных создается на основе выбранной модели организации данных целевой СУБД. Иначе говоря, на этом этапе уже должно быть известно, какая СУБД будет использоваться в качестве целевой - реляционная, сетевая, иерархическая или

объектно-ориентированная. Однако на этом этапе игнорируются все остальные характеристики выбранной СУБД, например, любые особенности физической организации ее структур хранения данных и построения индексов.

В процессе разработки логическая модель данных постоянно тестируется и проверяется на соответствие требованиям пользователей. Для проверки правильносги логической модели данных используется метод нормализации. Нормализация гарантирует, что отношения, выведенные из существующей модели данных, не будут обладать избыточностью данных, способной вызвать нарушения в процессе обновления данных после их физической реализации. Помимо всего прочего, логическая модель данных должна обеспечивать поддержку всех необходимых пользователям транзакций.

Созданная логическая модель данных является источником информации для этапа физического проектирования и обеспечивает разработчика физической базы данных средствами поиска компромиссов, необходимых для достижения поставленных целей, что очень важно для эффективного проектирования. Логическая модель данных играет также важную роль на этапе эксплуатации и сопровождения уже готовой системы. При правильно организованном сопровождении поддерживаемая в актуальном состоянии модель данных позволяет точно и наглядно представить любые вносимые в базу данных изменения, а также оценить их влияние на прикладные программы и использование данных, уже имеющихся в базе.

Физическое проектирование базы данных

Физическое проектирование базы данных. Процесс подготовки описания реализации базы данных на вторичных запоминающих устройствах; на этом этапе рассматриваются основные отношения, организация файлов и индексов, предназначенных для обеспечения эффективного доступа к данным, а также все связанные с этим ограничения целостности и средства защиты.

Физическое проектирование является третьим и последним этапом создания проекта базы данных, при выполнении которого проектировщик принимает решения о способах реализации разрабатываемой базы данных. Во время предыдущего этапа проектирования была определена логическая структура базы данных (которая описывает отношения и ограничения в рассматриваемой прикладной области). Хотя эта структура не зависит от конкретной целевой СУБД, она создается с учетом выбранной модели хранения данных, например реляционной, сетевой или иерархической. Однако, приступая к физическому проектированию базы данных, прежде всего необходимо выбрать конкретную целевую СУБД. Поэтому физическое проектирование неразрывно связано с конкретной СУБД. Между логическим и физическим проектированием существует постоянная обратная связь, так как решения, принимаемые на этапе физического проектирования с целью повышения производительности системы, способны повлиять на структуру логической модели данных.

Как правило, основной целью физического проектирования базы данных является описание способа физической реализации логического проекта базы данных. В случае реляционной модели данных под этим подразумевается следующее:

- создание набора реляционных таблиц и ограничений для них на основе информации, представленной в глобальной логической модели данных;
- определение конкретных структур хранения данных и методов доступа к ним, обеспечивающих оптимальную производительность СУБД;
- разработка средств защиты создаваемой системы.

Этапы концептуального и логического проектирования больших систем следует отделять от этапов физического проектирования. На это есть несколько причин.

- Они связаны с совершенно разными аспектами системы, поскольку отвечают на вопрос, что делать, а не как делать.
- Они выполняются в разное время, поскольку понять, что надо сделать, следует прежде, чем решить, как это сделать.
- Они требуют совершенно разных навыков и опыта, поэтому требуют привлечения специалистов различного профиля.

Определение требований к системе

Прежде чем перейти к проектированию приложения базы данных, важно установить задачи исследуемой системы и способы взаимодействия приложения с другими частями информационной системы организации. Эти задачи должны учитывать не только работу текущих пользователей и области применения разрабатываемой системы, но и будущих пользователей и другие возможные области применения. На рисунке приведена схема, которая определяет состав задач и область применения приложения базы данных для фирмы по продаже недвижимости. Кроме описания области применения приложения базы данных, необходимо определить основные пользовательские представления, которые поддерживаются базой данных.

Задачи системы для приложения базы данных

Пользовательские представления

В приложении базы данных может быть предусмотрено одно или несколько пользовательских представлений. Определение пользовательских представлений является существенной составляющей разработки приложения базы данных, поскольку позволяет гарантировать, чтобы ни одна важная категория пользователей базы данных не была исключена из рассмотрения при разработке требований к новому приложению. Пользовательские представления являются особенно полезными при создании относительно сложных приложений базы данных, поскольку позволяют разделить всю совокупность требований к базе данных на легко анализируемые группы.

Любое пользовательское представление определяет требования к приложению базы данных в части хранимых в ней данных и транзакций, выполняемых над данными (т.е. оно определяет, какие действия и над какими данными должен выполнять тот или иной пользователь). Требования пользовательского представления могут относиться только к данному представлению или частично совпадать с требованиями других представлений. На рисунке схематически изображена предметная область приложения базы данных с несколькими пользовательскими представлениями (которые обозначены цифрами 1-6). Обратите внимание, что требования некоторых пользовательских представлений (1—3, а также 5 и 6) частично перекрываются (это показано штриховкой), а требования пользовательского представления 4 являются индивидуальными.

Сбор и анализ требований пользователей

Проектирование базы данных основано на сборе и анализе информации о той части организации, которая будет обслуживаться базой данных. Сбор информации осуществляется для последующего создания основных пользовательских представлений (к ним относятся пользовательские представления для основных категорий пользователей или направлений деятельности предприятия). Ниже перечислена информация, которая требуется для решения указанной задачи.

- Описание применяемых или вырабатываемых данных.
- Подробные сведения о способах применения или выработки данных.

- Все дополнительные требования к создаваемому приложению базы данных.

Концептуальное проектирование базы данных

Концептуальное проектирование базы данных. Конструирование информационной модели предприятия, не зависящей от каких-либо физических условий реализации.

Концептуальное проектирование базы данных начинается с создания концептуальной модели данных предприятия, полностью независимой от любых деталей реализации. К последним относятся выбранный тип СУБД, состав программ приложения, используемый язык программирования, конкретная аппаратная платформа, вопросы производительности и любые другие физические особенности реализации.

Этапы концептуального проектирования:

1. **Создание локальной концептуальной модели данных исходя из представлений о предметной области каждого из типов пользователей.**

Охват предметной области данного предприятия.

2. **Определение типов сущностей.**

Определение основных типов сущностей, которые требуются для конкретного представления.

3. **Определение типов связей.**

Определение важнейших типов связей, существующих между сущностями, выделенными на предыдущем этапе.

4. **Определение атрибутов и связывание их с типами сущностей и связей.**

Связывание атрибутов с соответствующими типами сущностей или связей.

5. **Определение доменов атрибутов.**

Определение доменов для всех атрибутов, присутствующих в локальной концептуальной модели данных.

6. **Определение атрибутов, являющихся потенциальными и первичными ключами.**

Определение всех потенциальных ключей для каждого типа сущности и, если таких ключей окажется несколько, выбор среди них первичного ключа.

7. **Обоснование необходимости использования понятий расширенного моделирования (необязательный этап).**

Рассмотреть необходимость использования таких расширенных понятий моделирования, как уточнение/обобщение, агрегирование и композиция.

8. **Проверка модели на отсутствие избыточности.**

Проверка на отсутствие какой-либо избыточности данных в модели.

9. **Проверка соответствия локальной концептуальной модели конкретным пользовательским транзакциям.**

Убедиться в том, что локальная концептуальная модель поддерживает транзакции, необходимые для рассматриваемого представления.

10. **Обсуждение локальных концептуальных моделей данных с конечными пользователями.**

Обсуждение локальных концептуальных моделей данных с конечными пользователями с целью подтверждения того что данная модель полностью соответствует спецификации требований пользовательского представления.

Раздел 6. Проектирование БД.

В настоящей лекции определяется процесс проектирования базы данных и рассматривается базовая бизнес-модель процесс проектирования реляционной базы данных, основанная на понятии жизненного цикла.

Что такое проектирование базы данных

Значительная часть проектов в области информационных технологий (далее ИТ-проектов) направлена на разработку и создание информационных систем, в рамках которых осуществляется обработка данных различной сложности. Целью таких проектов является разработка и создание информационной системы с базами данных. Практически во всех таких проектах решается задача *проектирования баз данных* определенного типа. Решение задачи проектирования повышает *вероятность* того, что разрабатываемая информационная система (далее - система) будет удовлетворять заданным функциональным и информационным требованиям с учетом заданных ограничений.

Примеры *функциональных требований*: выдача отчетов *по* продажам *по* регионам; выдача отчетов *по* продажам *по* кварталам; автоматический расчет скидок на товары при увеличении объема закупаемой партии и т.п.

Примеры ограничений: максимальное время, отпущенное на проект; количество денежных средств, которое можно на него потратить. Следует также учитывать технологические средства, доступные при реализации проекта, например требование реализации *базы данных* в архитектуре "файл-*сервер*".

В эксплуатации *база данных* и ее окружение должны удовлетворять набору требований *по* ряду укрупненных (интегрированных) параметров, таких как:

- функциональность и адаптируемость;
- производительность обработки транзакций;
- пропускная способность;
- время реакции;
- безопасность.

Параметры, выражающие требования к базе данных, могут ранжироваться посредством присвоения приоритетов. Присвоение высшего приоритета требованию создать структуру данных для достижения системой максимально возможной производительности может привести к тому, что при *проектировании базы данных* требование обеспечить удобство работы *определенной категории* пользователей будет рассматриваться через призму производительности. Например, в системе бронирования авиабилетов в транснациональной авиакомпании время отклика на *запрос* не должно превышать 15-30 секунд. Поэтому, если это требование не будет удовлетворяться, то потребуется "разгрузить"*приложение* оператора.

Таким образом, процесс *проектирования базы данных* заключается в достижении компромиссов между функциональными, информационными, аппаратными, архитектурными и технологическими требованиями к базе данных и строится на информированном принятии решений *по структуре базы данных*.

Введем *определение проектирования баз данных*.

Определение 1. *Проектирование базы данных* - это поиск способов удовлетворения *функциональных требований* средствами имеющейся компьютерной технологии с учетом заданных ограничений.

Как правило, ИТ-проекты *по* созданию *базы данных* включают в себя следующие этапы: *определение* стратегии построения системы, *анализ требований* к базе данных, *проектирование базы данных*, реализация базы, тестирование и

внедрение *базы данных*. Этап *проектирования базы данных* считается одним из самых сложных "размытых" этапов *создания базы данных*, который не имеет явно выраженного начала и окончания.*По* сравнению с анализом требований к базе данных или разработкой приложений, *проектирование базы данных, по* мнению многих ведущих специалистов, является плохо структурированной задачей. Если все этапы *создания базы данных* перекрываются друг с другом в своей последовательности, то этап проектирования перекрывается со всеми остальными этапами. Проектирование начинается с момента принятия стратегических решений и продолжается на этапах реализации и тестирования.

Процесс *проектирования базы данных* охватывает несколько основных сфер.

- Проектирование объектов базы данных (таблицы, представления, индексы, триггеры, хранимые процедуры, функции, пакеты) для представления данных предметной области в базе данных.
- Проектирование интерфейса взаимодействия с базой данных (формы, отчеты и т.д.), т.е. проектирование приложений, которые будут сопровождать данные в базе данных и реализовывать вопросно-ответные отношения на этих данных.
- *Проектирование баз данных* под конкретную вычислительную среду или информационную технологию (архитектура "клиент-сервер", параллельные архитектуры, распределенная вычислительная среда).
- *Проектирование баз данных* под назначение системы (*интеллектуальный анализ данных, OLAP, OLTP* и т.д.).

Техника *проектирования баз данных* может измениться в целом и в деталях в зависимости от назначения системы. Например, следует различать проектирование систем складирования данных и проектирование так называемых *OLTP* - систем, ориентируемых на оперативную обработку транзакций. В данном учебном курсе рассматривается *проектирование баз данных* в основном для *OLTP* -систем. Именно на таких системах исторически сложилась техника *проектирования баз данных*.

Известно, что *база данных*:

- имеет свою внутреннюю архитектуру;
- имеет свое собственное лингвистическое содержание;
- действует в рамках некоторой внешней среды;
- имеет свои средства взаимодействия с внешней средой;
- функционирует на конкретной программно-аппаратной платформе;
- поддерживается в рамках определенных организационно-технологических мероприятий.

Типовая бизнес-модель процесса проектирования базы данных

Процесс *проектирования базы данных* может быть представлен в виде модели бизнес-процессов. Обычно проектировщики не создают *бизнес-модель* процесса *проектирования базы данных*. А напрасно! *Бизнес-модель* процесса проектирования позволяет:

- отобразить субъективное мнение проектировщика баз данных на процесс проектирования конкретной базы данных;
- учесть особенности ИТ-проекта, в рамках которого проектируется база данных;
- достаточно быстро составить план проектирования конкретной базы данных;
- просчитать длительность проектных работ (создать временную *модель проектирования*).

Рассмотрим типовую *бизнес-модель* процесса *проектирования базы данных*. На рис. 3.1 приведена *контекстная диаграмма* процесса *проектирования базы данных*.

Рис. 3.1. Контекстная диаграмма процесса проектирования базы данных

Как видно из рисунка, на вход процесса *проектирования базы данных* подаются:

- *информационная модель предметной области* базы данных: диаграммы "*сущность-связь*" (ER -диаграммы);
- *функциональная модель предметной области* базы данных: *бизнес-модель* процессов, *диаграммы потока данных* (DF -диаграммы),*диаграммы состояний*, - *диаграммы жизненных циклов* сущностей, спецификации на системы (требования), бизнес-правила;
- общесистемные требования и ограничения;
- задачи обратного влияния.

Могут быть представлены и другие документы.

Примечание. Под задачами обратного влияния здесь понимается совокупность проблем, которые возникают в процессе разработки приложений базы данных, ее тестирования, опытной и промышленной эксплуатации и приводят к модификации*физической модели* базы данных. Примером такой задачи является настройка операторов SELECT с целью увеличения производительности выборки.

На выходе процесса *проектирования базы данных* формируются следующие результаты:

- физическая модель базы данных, которая может быть преобразована в скрипт для *создания базы данных*;
- физическая база данных;
- спецификация модулей приложений базы данных;
- план тестирования базы данных.

По требованию может быть разработана и другая документация.

Продолжим функциональную декомпозицию процесса *проектирования базы данных*. На рис. 3.2 приведена *диаграмма декомпозиции* процесса*проектирования базы данных* первого уровня, отражающая основные наиболее крупные профессиональные задачи (этапы) *проектирования базы данных*.

Внимание! В настоящем курсе рассматривается минимально необходимый, с точки зрения автора, набор задач, позволяющий спроектировать реляционную базу данных.

Рис. 3.2. Диаграмма декомпозиция процесса проектирования базы данных: первый уровень

Такими задачами (этапами) являются:

- сбор и анализ входных данных;
- *создание логической модели базы данных* ;
- *создание физической модели базы данных*: *внутренняя схема*;
- *создание физической модели базы данных*: *учет влияния транзакций* ;
- *создание серверного кода* ;
- *проектирование модулей приложений* базы данных;
- *контроль качества проектирования базы данных* ;
- задачи обратного влияния.

Сбор и анализ входных данных - это начальный этап проектирования, на котором осуществляется сбор и *контроль* качества результатов*анализа предметной области базы данных*, готовится план *проектирования базы данных*.

Создание логической модели базы данных - это этап, на котором на основании *информационной модели предметной области базы данных*создается логическая *структура базы данных*, независимая от ее реализации.

Создание физической модели базы данных: *внутренняя схема* - это этап, на котором на основании логической модели *базы данных* создается физическая *структура базы данных*, зависимая от ее реализации. На этом этапе выполняется преобразование отношений *логической модели реляционной базы данных* в команды создания объектов физической *базы данных*, в результате чего создается так называемая *внутренняя схема базы данных*. Дополнительно может быть создана так называемая внешняя *схема базы данных*, которая отражает точку зрения пользователей на данные в базе данных. Полученный *скрипт* может быть применен для создания физической *базы данных*.

Создание физической модели базы данных: *учет влияния транзакций* - это этап, на котором анализируются возможные транзакции системы, выполняется, в случае необходимости, *денормализация* отношений для обеспечения более высокой производительности *базы данных*. На этом этапе создается *скрипт* создания физической *базы данных*.

Создание серверного кода - это этап, на котором на основании *функциональной модели предметной области базы данных* создается серверный код *базы данных* в виде триггеров, хранимых процедур и пакетов. Эти модули создаются проектировщиком *базы данных* и выполняются сервером.

Проектирование модулей приложений - это этап, на котором создаются *спецификации модулей* приложений, разрабатываются стратегии тестирования *базы данных* и приложений, создается план тестирования приложений *базы данных* и готовятся тестовые данные.

Контроль качества *проектирования базы данных* заключается в проверке качества результатов проектирования на каждом его этапе.

Учет задач обратного влияния заключается в настройке некоторых транзакций к базе данных и локальном перепроектировании *базы данных*согласно требованиям, поступающим с других этапов *создания базы данных*.

Как правило, последние четыре из сформулированных нами задач решаются в соответствии с правилами и стандартами, принятыми в конкретной организации.

Поэтому мы не будем строить для них детальной *бизнес-модели* в настоящей лекции, а в лекциях, посвященных рассмотрению этих задач, мы ограничимся понятийным материалом и установочными рекомендациями. В частности, задача контроля качества *проектирования базы данных*упомянута в настоящем курсе для полноты представления материала и как самостоятельная задача в настоящих лекциях не изучается. Это предмет отдельного курса лекций.

Бизнес-модель процесса проектирования базы данных: сбор и анализ входных данных

Перейдем ко второму уровню функциональной декомпозиции процесса *проектирования базы данных* в рамках его первого этапа - *сбор и анализ входных данных*. На рис. 3.3 представлена *диаграмма декомпозиции* процесса *проектирования базы данных* второго уровня, отражающая основные задачи этапа сбора и анализа входных данных.

Рис. 3.3. Диаграмма декомпозициии процесса проектирования базы данных: второй уровень. Сбор и анализ входных данных

Такими задачами являются:

- сбор документации с результатами *анализа предметной области* базы данных в виде диаграмм, спецификаций и требований;
- контроль качества результатов *анализа предметной области* базы данных;

- систематизация требований и спецификаций заказчика к базе данных;
- подготовка плана *проектирования базы данных*.

Бизнес-модель процесса проектирования реляционной базы данных: создание логической модели базы данных

Продолжим функциональную декомпозицию процесса *проектирования реляционной базы данных* в рамках его второго этапа - *создания логической модели базы данных*.

Основной целью этапа *создания логической модели базы данных* является преобразование *информационной модели предметной области базы данных* в логическую модель реляционной *базы данных*. Создание логической модели *базы данных* предполагает решение следующих основных задач и выполнения операций в рамках таких задач:

- нормализация *сущностей предметной области*:
  - получить список *атрибутов сущности*;
  - определить *функциональные зависимости* (ФЗ) в сущности;
  - определить *детерминанты* сущности;
  - определить *возможные ключи* отношения, в частности, рассмотрев *уникальный идентификатор сущности*.
  - выполнить нормализацию сущности (преобразовать сущность в отношение);
  - для полученного отношения назначить первичные ключи;
  - сформировать список кандидатов на внешние ключи, если необходимо;
  - сформировать бизнес-правила поддержки *целостности сущности*, если необходимо;
- нормализация отношений логической модели базы данных:
- определить *степень связи* сущностей;
- определить *класс принадлежности сущности* к связи;
  - нормализовать отношение (разрешить связи);
  - назначить первичные ключи связывающих отношений, исходя из уникального *идентификатора связи* и процедуры миграции ключей при нормализации;
  - определить атрибуты связывающих отношений, если необходимо;
  - сформировать бизнес-правила поддержки целостности связей;
- проверка правильности *логической модели реляционной базы данных*:
  - проверка отношений на соответствие *нормальной форме Бойса-Кодда*;
  - проверка отношений на свойства соединения без потерь и сохранения *функциональных зависимостей*;
  - предотвращение потери данных путем миграции *первичных ключей отношения* и назначения внешних ключей;
  - проверка на отсутствие незамкнутых связей;
  - проверка на отсутствие одиночных отношений;
- формулировка части исходных данных для решения задачи управления ссылочной целостностью;
- документирование *логической модели реляционной базы данных*;
- принятие решения о реализуемости построенной *логической модели реляционной базы данных*;
- принятие решения о разработке *физической модели реляционной базы данных*.

Результатом проектирования логической модели *базы данных* является нормализованная *схема отношений базы данных*. Отметим, что в ходе выполнения этапа *создания логической модели базы данных* могут быть созданы новые объекты *базы данных*, не предусмотренные в*информационной модели предметной области*, например *связывающая сущность* при нормализации отношения со степенью связи "*многие-ко-многим*". Иногда на этом этапе принимается решение о выборочной *денормализации* отношений.

На рис. 3.4-рис. 3.6 представлены *бизнес-модели* процессов *создания логической модели базы данных*, нормализации *сущности предметной области* и нормализации отношений логической модели *базы данных* соответственно.

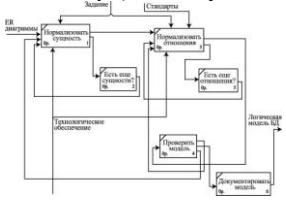


Рис. 3.4. Бизнес-модель процесса создания логической модели базы данных

Рис. 3.5. Бизнес-модель процесса нормализации сущности

Рис. 3.6. Бизнес-модель процесса нормализации отношения

Представленные задачи составляют минимально необходимый набор задач, позволяющих спроектировать логическую модель *базы данных*, и могут рассматриваться как один из возможных способов организации *работ* в этой области.

Методология концептуального проектирования реляционных баз данных

Концептуальное проектирование базы данных. Конструирование информационной модели предприятия, не зависящей от каких-либо физических условий реализации.

Концептуальное проектирование базы данных начинается с создания концептуальной модели данных предприятия, полностью независимой от любых деталей реализации. К последним относятся выбранный тип СУБД, состав программ приложения, используемый язык программирования, конкретная аппаратная платформа, вопросы производительности и любые другие физические особенности реализации.

Этапы концептуального проектирования:

1. **Создание локальной концептуальной модели данных исходя из представлений о предметной области каждого из типов пользователей.**

Охват предметной области данного предприятия.

2. **Определение типов сущностей.**

Определение основных типов сущностей, которые требуются для конкретного представления.

3. **Определение типов связей.**

Определение важнейших типов связей, существующих между сущностями, выделенными на предыдущем этапе.

4. **Определение атрибутов и связывание их с типами сущностей и связей.**

Связывание атрибутов с соответствующими типами сущностей или связей.

5. **Определение доменов атрибутов.**

Определение доменов для всех атрибутов, присутствующих в локальной концептуальной модели данных.

6. **Определение атрибутов, являющихся потенциальными и первичными ключами.**

Определение всех потенциальных ключей для каждого типа сущности и, если таких ключей окажется несколько, выбор среди них первичного ключа.

7. **Обоснование необходимости использования понятий расширенного моделирования (необязательный этап).**

Рассмотреть необходимость использования таких расширенных понятий моделирования, как уточнение/обобщение, агрегирование и композиция.

8. **Проверка модели на отсутствие избыточности.**

Проверка на отсутствие какой-либо избыточности данных в модели.

9. **Проверка соответствия локальной концептуальной модели конкретным пользовательским транзакциям.**

Убедиться в том, что локальная концептуальная модель поддерживает транзакции, необходимые для рассматриваемого представления.

10. **Обсуждение локальных концептуальных моделей данных с конечными пользователями.**

Обсуждение локальных концептуальных моделей данных с конечными пользователями с целью подтверждения того что данная модель полностью соответствует спецификации требований пользовательского представления.

Методология логического проектирования реляционных баз данных

Логическое проектирование базы данных - процесс создания модели используемой на предприятии информации на основе выбранной модели организации данных, но без учета типа целевой СУБД и других физических аспектов реализации.

Логическое проектирование является вторым этапом проектирования базы данных, на котором происходит уточнение и преобразование концептуальной модели, созданной на предыдущем этапе, в логическую модель данных. Логическая модель данных создается на основе выбранной модели организации данных целевой СУБД (реляционная, сетевая, иерархическая или объектно-ориентированная), но этапе игнорирует все остальные характеристики выбранной СУБД, например, любые особенности физической организации ее структур хранения данных и построения индексов.

В процессе разработки логическая модель данных постоянно тестируется и проверяется на соответствие требованиям пользователей. Для проверки правильности логической модели данных используется метод нормализации, который гарантирует, что отношения, выведенные из существующей модели данных, не будут обладать избыточностью данных, способной вызвать нарушения в процессе обновления данных после их физической реализации.

Созданная логическая модель данных является источником информации для этапа физического проектирования и предоставляет разработчику физической модели данных средства проведения всестороннего анализа различных аспектов работы с данными, что имеет исключительно важное значение для выбора действительно эффективного проектного решения. Логическая модель данных играет также важную роль на этапе эксплуатации и сопровождения уже готовой системы. При правильно организованном сопровождении поддерживаемая в актуальном состоянии модель данных позволяет точно и наглядно представить любые вносимые в базу данных изменения, а также оценить их влияние на прикладные программы и использование данных, уже имеющихся в базе.

Нормализация и нормальные формы

Нормализация - метод создания набора отношений с заданными свойствами на основе требований к данным, установленных в некоторой организации.

Дополнительно:

Процесс нормализации был впервые предложен Э. Ф. Коддом. Нормализация часто выполняется в виде последовательности тестов с целью проверки соответствия (или несоответствия) некоторого отношения требованиям заданной нормальной формы. Сначала были предложены только три вида нормальных форм: первая (1НФ), вторая (2НФ) и третья (3НФ). Затем Р. Бойсом и Э. Ф. Коддом было сформулировано более строгое определение третьей нормальной формы, которое получило название нормальной формы Бойса-Кодда (НФБК). Все эти нормальные формы основаны на функциональных зависимостях, существующих между атрибутами отношения, Нормальные формы более высокого порядка, которые превосходят НФБК, были введены позднее. К ним относятся четвертая (4НФ) и пятая (5НФ) нормальные формы. Но на практике эти нормальные формы более высоких порядков используются крайне редко.

Отношение было представлено как состоящее из некоторого количества атрибутов, а реляционная схема — из некоторого количества отношений. Атрибуты могут группироваться в отношения с образованием реляционной схемы на основе либо собственного опыта разработчика базы данных, либо посредством вывода реляционной схемы из

разработанной ER-диаграммы. При использовании любого из этих двух подходов часто требуется применять определенный формальный метод, способный помочь проектировщику базы данных найти оптимальную группировку атрибутов для каждого отношения в схеме.

Процесс нормализации является формальным методом, позволяющим определять отношения на основе их первичных или потенциальных ключей и функциональных зависимостей, существующих между их атрибутами. Проектировщики баз данных могут использовать нормализацию в виде наборов тестов, применяемых к отдельным отношениям с целью нормализации реляционной схемы до заданной конкретной формы, что позволит предотвратить возможное возникновение аномалий обновления.

Этапы методологии физического проектирования баз данных:

1. Перенос глобальной логической модели данных в среду целевой СУБД.
2. Проектирование основных отношений.
3. Разработка способов получения производных данных.
4. Реализация ограничений предметной области.
5. Проектирование физического представления базы данных.
6. Анализ транзакций.
7. Выбор файловой структуры.
8. Определение индексов.
9. Определение требований к дисковой памяти.
10. Проектирование пользовательских представлений.
11. Разработка механизмов защиты.
12. Обоснование необходимости введения контролируемой избыточности.
13. Текущий контроль и настройка операционной системы.

Проектирование физического представления базы данных

Определение оптимальной файловой структуры для хранения базовых отношений и индексов, необходимых для достижения приемлемой производительности. Иными словами, определение способа хранения отношений и кортежей во вторичной памяти.

Анализ транзакций

Определение функциональных характеристик транзакций, которые будут выполняться в проектируемой базе данных, и выделение наиболее важных из них.

Выбор файловой структуры

Определение наиболее эффективной файловой структуры для каждого базового отношения.

Выбор индексов

Определение того, будет ли добавление индексов способствовать повышению производительности системы.

Определение требований к дисковому пространству

Оценка объема дискового пространства, необходимого для размещения базы данных.

Раздел 7. Структурированный язык запросов SQL. . (комп. презентация – 5ч.)

SQL - язык манипулирования данными

Назначение языка SQL

Любой язык работы с базами данных должен предоставлять пользователю следующие возможности:

- создавать базы данных и таблицы с полным описанием их структуры;
- выполнять основные операции манипулирования данными, такие как вставка, модификация и удаление данных из таблиц;
- выполнять простые и сложные запросы.

Кроме того, язык работы с базами данных должен решать все указанные выше задачи при минимальных усилиях со стороны пользователя, а структура и синтаксис его команд должны быть достаточно просты и доступны для изучения. И, наконец, он должен быть универсальным, т.е. отвечать некоторому признанному стандарту, что позволит использовать один и тот же синтаксис и структуру команд при переходе от одной СУБД к другой. Язык SQL удовлетворяет практически всем этим требованиям.

SQL является примером языка преобразования данных, или же языка, предназначенного для работы с таблицами с целью преобразования входных данных к требуемому выходному виду. Язык SQL, который определен стандартом ISO, имеет два основных компонента:

- язык DDL (Data Definition Language), предназначенный для определения структур базы данных и управления доступом к данным;
- язык DML (Data Manipulation Language), предназначенный для выборки и обновления данных.

До появления стандарта SQL3 язык SQL включал только команды определения и манипулирования данными; в нем отсутствовали какие-либо команды управления ходом вычислений. Другими словами, в этом языке не было команд IF ... THEN ... ELSE, GO TO, DO ... WHILE и любых других, предназначенных для управления ходом вычислительного процесса. Подобные задачи должны были решаться программным путем (с помощью языков программирования или управления заданиями) либо интерактивно (в результате действий, выполняемых самим пользователем). По причине подобной незавершенности (с точки зрения организации вычислительного процесса) язык SQL мог использоваться двумя способами. Первый предусматривал интерактивную работу, заключающуюся во вводе пользователем с терминала отдельных операторов SQL. Второй состоял во внедрении операторов SQL в программы на процедурных языках.

Язык SQL относительно прост в изучении.

- Это непроцедурный язык, поэтому в нем необходимо указывать, какая информация должна быть получена, а не как ее можно получить. Иначе говоря, язык SQL не требует указания методов доступа к данным,
- Как и большинство современных языков, SQL поддерживает свободный формат записи операторов. Это означает, что при вводе отдельные элементы операторов не связаны с фиксированными позициями на экране.
- Структура команд задается набором ключевых слов, представляющих собой обычные слова английского языка, такие как CREATE TABLE (Создать таблицу), INSERT (Вставить), SELECT (Выбрать).

Например:

```
CREATE TABLE Staff (staffNo VARCHAR(S), IName VARCHAR(15), salary DECIMAL(7,2));
INSERT INTO Staff VALUES ('SG16', 'Brown', 8300);
```

```
SELECT staffNo, IName, salary
FROM Staff
WHERE salary > 10000;
```

- Язык SQL может использоваться широким кругом пользователей, включая администраторов баз данных (АБД), руководящий персонал компании, прикладных программистов и множество других конечных пользователей разных категорий.

В настоящее время для языка SQL существуют международные стандарты, формально определяющие его как стандартный язык создания и манипулирования реляционными базами данных, каковым он фактически и является.

4.3. Лабораторные работы

№ п/п	*Номер раздела дисциплины*	*Наименование тем лабораторных работ*	*Объем (час.)*	*Вид занятия в интерактивной, активной, инновационной формах, (час.)*
1.	**1,2,3**	Создание таблиц в ручном режиме средствами СУБД	3	-
2.	**1,2,3**	Создание таблиц базы данных в автоматизированном режиме средствами СУБД.	3	-
3.	**3.**	Создание и использование запросов	3	-
4.	**3.**	Создание экранных форм и отчетов	3	3
5.	**4.**	Определение сущностей и атрибутов предметной области.	4	-
6.	**4.**	Нормализация данных	3	3
7.	**1,2,3,5, 6**	Создание базы данных	3	-
8.	**7.**	Создание SQL-запросов	4	3
9.	**7.**	Создание группирующих и перекрестных запросов на SQL	3	-
10.	**7.**	Создание запросов действия и перекрестных запросов на SQL	3	-
11.	**5,6**	Создание клиентского приложения	4	3
		ИТОГО	**36**	**12**

4.4. Практические занятия

Учебным планом не предусмотрено

4.5. Курсовая работа

Цель: закрепление теоретических и практических знаний, полученных при изучении дисциплины «Базы данных». При выполнении курсовой работы обучающийся должен приобрести практические навыки:

- в освоении методологии нормализации отношений при создании схемы базы данных для конкретной предметной области;
- проектирования БД;
- по созданию баз данных, схем данных, заполнению и редактированию таблиц данных в выбранной СУБД;
- реализации запросов на SQL.

Курсовая работа выполняется в виде пояснительной записки объемом 20-25 страниц, оформляется в строгом соответствии со стандартом ФГБОУ ВО «БрГУ».

Структура работы:

- титульный лист;

- содержание;
- введение;
- основные разделы работы;
- заключение;
- список использованных источников;
- приложения.

Выдача задания, прием кр/Р и защита КП (КР) проводится в соответствии с календарным учебным графиком

Оценка	Критерии оценки курсовой работы
отлично	соответствие требованиям по структурному содержанию и объему работы; правильность выполнения задания, сопровождающегося выводами, рисунками, таблицами, диаграммами; правильность решения практических заданий, самостоятельность выполнения; оформление работы и списка использованных источников соответствует требованиям СТП 1.4-01-2005; грамотность, отсутствие стилистических ошибок; уверенное владение материалом при устной защите.
хорошо	соответствие требованиям по структурному содержанию и объему работы; правильность выполнения задания, сопровождающегося рисунками, таблицами, диаграммами; небольшие неточности при выполнении практической части работы, самостоятельность выполнения; оформление работы и списка использованных источников соответствует требованиям СТП 1.4-01-2005; грамотность, отсутствие стилистических ошибок; владение материалом при устной защите.
удовлетворительно	соответствие требованиям по структурному содержанию и объему работы; выполнение задания, не вв полном объеме сопровождается выводами, рисунками, таблицами, диаграммами; ошибки при выполнении практической части работы, самостоятельность выполнения; оформление работы и списка использованных источников соответствует требованиям СТП 1.4-01-2005; неуверенное владение материалом при устной защите.
неудовлетворительно	несоответствие требованиям по структурному содержанию и объему работы; неправильность выполнения задания, сопровождающегося рисунками, таблицами, диаграммами; наличие ошибок в выполнении практических заданий; отсутствие самостоятельности выполнения; оформление работы и списка использованных источников не соответствует требованиям СТП 1.4-01-2005; наличие стилистических ошибок; отсутствие владения материалом при устной защите.

5. МАТРИЦА СООТНЕСЕНИЯ РАЗДЕЛОВ УЧЕБНОЙ ДИСЦИПЛИНЫ К ФОРМИРУЕМЫМ В НИХ КОМПЕТЕНЦИЯМ И ОЦЕНКЕ РЕЗУЛЬТАТОВ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Компетенции / *№, наименование разделов дисциплины*	*Кол-во часов*	*Компетенции*				Σ *комп.*	$t_{ср}$, *час*	*Вид учебной работы*	*Оценка результатов*
		ОПК	*ПК*						
		1	***14***	***6***	***7***				
1	2	3	4	5	6	7	8	9	10
1. Основные положения теории БД.	8	-	+	-	-	1	8	Лк, ЛР, СРС	зачет
2. Модели данных.	6	-	+	-	-	1	6	Лк, ЛР, КР, СРС	зачет
3. Реляционные модели и языки.	19	-	+	-	-	1	19	Лк, ЛР, КР, СРС	зачет
4. Предметная область базы данных.	45	-	-	+	-	1	45	Лк, ЛР, КР, СРС	экзамен
5. Жизненный цикл БД.	15	+	+	-	-	1	7,5	Лк, ЛР, КР, СРС	экзамен
6. Проектирование БД.	65	-	-	-	+	2	65	Лк, ЛР, КР, СРС	экзамен
7. Структурированный язык запросов SQL.	31	-	+	-	-	1	31	Лк, ЛР, КР, СРС	экзамен
всего часов	**189**	**7,5**	**71,5**	**45**	**65**	**4**	**189**		

6. ПЕРЕЧЕНЬ УЧЕБНО-МЕТОДИЧЕСКОГО ОБЕСПЕЧЕНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

1. Вахрушева М.Ю. Базы данных: методические указания по выполнению лабораторных работ. – В 2ч. – Братск: Изд-во БрГУ, 2014. –Ч.1. – 52с.
2. Вахрушева М.Ю. Базы данных: методические указания по выполнению лабораторных работ. – В 2ч. – Братск: Изд-во БрГУ, 2014. –Ч.2. – 71с.
3. Вахрушева М.Ю. Базы данных: методические указания по выполнению курсовой работы. – Братск : Изд-во БрГУ, 2014. - 58с.
4. Шичкина Ю.А. Разработка приложений для работы с базами данных в среде программирования VisualStudio C#. В 2 ч. Ч.1,2 / Ю. А. Шичкина, В. С. Кедрин . - Братск : БрГУ, 2013.Ч.1 : Базы данных на базе SQL serverCompact 3.5. - 2013. - 100 с.
5. Шичкина Ю.А. Разработка приложений для работы с базами данных в среде программирования VisualStudio C#. В 2 ч. Ч.1,2 / Ю. А. Шичкина. - Братск :БрГУ, 2013. Ч.2 : Клиент-серверные и XML-ориентированные базы данных с доступом к данным ASP.NET. - 156 с.

7. ПЕРЕЧЕНЬ ОСНОВНОЙ И ДОПОЛНИТЕЛЬНОЙ ЛИТЕРАТУРЫ, НЕОБХОДИМОЙ ДЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ

№	***Наименование издания*** *(автор, заглавие, выходные данные)*	***Вид занятия*** (*Лк, ПЗ, КР, СР)*	***Количество экземпляров в библиотеке*, шт.**	***Обеспеченность*, (экз./ чел.)**
1	**2**	**3**	**4**	**5**
	Основная литература			
1	Карпова, Т.С. Базы данных: модели, разработка, реализация : учебное пособие / Т.С. Карпова. - 2-е изд., исправ. - Москва : Национальный Открытый Университет «ИНТУИТ», 2016. - 241 с. : ил. ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=429003			
2	Советов Б.Я. Базы данных: теория и практика : учебник для бакалавров / Б.Я. Советов, В. В. Цехановский, В. Д. Чертовской. - 2-е изд. - М. : Юрайт, 2014. - 463 с. - (Бакалавр. Базовый курс).	Лк, ЛЗ, КР, СР	15	1
3	Кузнецов С.Д. Базы данных : учебник / С.Д. Кузнецов. - М. Академия, 2012. - 496 с. - (Прикладная математика и информатика).	Лк,ЛЗ, КР, СР	15	1
4	Хомоненко А.Д. Базы данных: учебник для вузов по техническим и экономическим специальностям / А.Д. Хомоненко, В.М. Цыганков, М.Г. Мальцев; Ред. А.Д. Хомоненко. – 5-е изд. – М.: БИНОМ-Пресс, 2006. – 736 с.	Лк,ЛЗ, КР, СР	15	1
	2. Дополнительная литература			
5	Системы управления базами данных : лабораторный практикум / сост. Д.Л. Осипов, М.Г. Огур ; Министерство образования и науки РФ, Федеральное государственное автономное образовательное учреждение высшего образования «Северо-Кавказский федеральный университет». - Ставрополь : СКФУ, 2017. - 148 с. - Библиогр. в кн. ; То же [Электронный ресурс]. - URL:	Лк, ЛЗ, КР, СР	ЭР (1)	1

	http://biblioclub.ru/index.php?page=book&id=483760			
6	Вахрушева М.Ю. Базы данных: методические указания по выполнению лабораторных работ. – В 2ч. – Братск: Изд-во БрГУ, 2014. –Ч.1. – 52с.	ЛЗ,СР	45	1
7	Вахрушева М.Ю. Базы данных: методические указания по выполнению лабораторных работ. – В 2ч. – Братск: Изд-во БрГУ, 2014. –Ч.2. – 71с.	ЛЗ, СР	45	1
8	Вахрушева М.Ю. Базы данных: методические указания по выполнению курсовой работы. – Братск : Изд-во БрГУ, 2014. - 58с.	КР, СР	48	1
9	Гринченко Н.Н. Проектирование баз данных. СУБД MicrosoftAccess: учебное пособие / Н. Н. Гринченко [и др.]. - 2-е изд., стереотип. - Москва : Горячая линия-Телеком, 2013.	ЛЗ, КР,СР	10	1
10	Заботина Н. Н. Проектирование информационных систем : учебное пособие / Н. Н. Заботина. - М. : ИНФРА-М, 2013. - 331 с. - (Высшее образование: Бакалавриат).	Лк, ЛЗ, КР, СР	3	0,3
11	Шичкина Ю.А. Разработка приложений для работы с базами данных в среде программирования VisualStudio C#. В 2 ч. Ч.1,2 / Ю. А. Шичкина, В. С. Кедрин . - Братск : БрГУ, 2013. Ч.1 : Базы данных на базе SQL serverCompact 3.5. - 2013. - 100 с.	ЛЗ, КР, СР	90	1
12	Шичкина Ю.А. Разработка приложений для работы с базами данных в среде программирования VisualStudio C#. В 2 ч. Ч.1,2 / Ю. А. Шичкина. - Братск :БрГУ, 2013. Ч.2 : Клиент-серверные и XML-ориентированные базы данных с доступом к данным ASP.NET. - 156 с.	ЛЗ, КР, СР	91	1
13	Голицына О.Л. Базы данных : учебное пособие / О. Л. Голицына, Н. В. Максимов, И. И. Попов. - 3-е изд., перераб. и доп. - М. : Форум, 2012. - 400 с. - (Высшее образование: Бакалавриат).	Лк, ЛЗ, КР, СР	15	1

8. ПЕРЕЧЕНЬ РЕСУРСОВ ИНФОРМАЦИОННО - ТЕЛЕКОММУНИКАЦИОННОЙ СЕТИ «ИНТЕРНЕТ» НЕОБХОДИМЫХ ДЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ

1. Электронный каталог библиотеки БрГУ: http://irbis.brstu.ru/CGI/irbis64r_15/cgiirbis_64.exe?LNG=&C21COM=F&I21DBN=BOOK&P21DBN=BOOK&S21CNR=&Z21ID=
2. Электронная библиотека БрГУ http://ecat.brstu.ru/catalog
3. Федеральная университетская компьютерная сеть России // Электронный ресурс [Режим доступа: свободный] http://www.runnet.ru/
4. Каталог учебников, оборудования, электронных ресурсов // Электронный ресурс [Режим доступа: свободный] http://ndce.edu.ru/
5. Электронно-библиотечная система издательства «Лань» // Электронный ресурс http://e.lanbook.com/ ,
6. Библиотека «Книгосайт» // Электронный ресурс [Режим доступа: свободный] http://knigosite.ru/
7. Электронная библиотека книг на тему бизнеса, финансов, экономики и смежным темам // Электронный ресурс [Режим доступа: свободный] http://www.finbook.biz/
8. ЭБС «Университетская библиотека online» // Электронный ресурс http://biblioclub.ru/ ,

9. Научная электронная библиотека «КИБЕРЛЕНИНКА» // Электронный ресурс [Режим доступа: свободный] http://cyberleninka.ru/

9. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

Вид учебных занятий	Организация деятельности обучающихся
Лекции	Написание конспекта лекций: кратко, последовательно фиксировать основные положения, выводы, формулировки, обобщения; помечать важные мысли, выделять ключевые слова, термины. Проверка терминов с помощью энциклопедий, словарей, справочников с выписыванием толкований в тетрадь. Обозначить вопросы, термины, материал, который вызывает трудности, пометить и попытаться найти ответ в рекомендуемой литературе. Если самостоятельно не удается разобраться в материале, необходимо сформулировать вопрос и задать преподавателю на консультации, практическом занятии.
Лабораторные занятия	Развитие интеллектуальных умений, практических навыков, подготовка ответов к контрольным вопросам, работа с основной и дополнительной литературой, необходимой для освоения дисциплины, выполнение заданий, активное участие в интерактивной, активной, инновационной формах обучения, составление и оформление отчетов по лабораторным работам.
Курсовая работа	Работа с основной и дополнительной литературой, необходимой для выполнения курсовой работы, углубление и конкретизация необходимого в соответствии с темой материала из литературных источников и полученных теоретических знаний, выработка способности и готовности их использования в практической исследовательской работе. Развитие интеллектуальных умений изложения материала, представления с элементами визуализации (схемы, графики, рисунки, таблицы, формулы) и оформления в соответствии с требованиями ГОСТ.
Самостоятельная работа обучающихся	*Подготовка к практическим занятиям.* Проработка основной и дополнительной литературы, терминов, сведений, требующихся для запоминания и являющихся основополагающими в теме/разделе. Конспектирование прочитанных литературных источников. Проработка материалов по изучаемому вопросу, с использованием на рекомендуемых ресурсов информационно-телекоммуникационной сети «Интернет». Выполнение заданий преподавателя, необходимых для подготовки к участию в интерактивной, активной, инновационных формах обучения по изучаемой теме. *Подготовка к экзамену.* При подготовке к экзамену необходимо ориентироваться на конспекты лекций, рекомендуемую литературу, использовать рекомендуемые ресурсы информационно-телекоммуникационной сети «Интернет».

9.1. Методические указания для обучающихся по выполнению лабораторных работ

Лабораторная работа № 1. Создание таблиц в ручном режиме средствами СУБД

Цель:

1. Научиться создавать таблицы базы данных в режиме **Конструктор.**
2. Освоить *основные приемы* заполнения и редактирования таблицы базы данных.
3. Познакомиться с простой сортировкой значений таблицы и с поиском записей по образцу.
4. Научиться создавать таблицы базы данных **в режиме таблицы**.
5. Научиться создавать самостоятельно *ключевые поля.*

Создание однотабличной базы данных

1.1. В меню **Пуск** выбрать команду **Программы/ MS Access**.
1.2. В окне диалога **MS Access** включить переключатель **Новая база данных** и щелкнуть по кнопке **ОК.**
1.3. В появившемся окне **Файл новой базы данных** выбрать вкладку **Мои документы,** а в поле **Имя файла** ввести *свою фамилию* и щелкнуть по кнопке **Создать.**
1.4. В окне **Имя БД: база данных** выбрать объект Т**аблицы** и щелкнуть по кнопке **Создать**.
1.5. В окне **Новая таблица** выбрать вариант создания таблицы **Конструктор** и щелкнуть по кнопке **ОК**.
1.6.В окне **Таблица 1: таблица** заполнить поля в соответствии с табл.1:

Таблица 1

Имя поля	**Тип данных**
Фамилия	Текстовый
Имя	Текстовый
Отчество	Текстовый
Год рождения	Числовой
Школа	Числовой
Класс	Числовой

Примечания:
– в поле **Имя поля** ввести слово *«Фамилия»*, а не свою фамилию;

– тип данных выбирать с помощью ниспадающего меню.

1.7.Сохранить таблицу, щелкнув по кнопке **Сохранить** на панели инструментов.

1.8. В появившемся окне **Сохранение** набрать *имя таблицы* **Список** и щелкнуть по кнопке **ОК**. Появится запрос на создание ключевого поля – нажать кнопку **ДА**.

1.9. Перейти в режим **Таблица**, щелкнув по кнопке **Вид/Режим таблицы** *на панели инструментов*.

1.10. В открывшемся окне **Список: таблица** заполнить клетки в соответствии с табл.2; при этом:

– значение поля **Код** будет меняться *автоматически;*

– ввод в ячейку заканчивать нажатием на клавишу **Enter.**

Таблица 2

Код	Фамилия	Имя	Отчество	Год рождения	Школа	Класс
1	Иванникова	Анна	Ивановна	1984	1	9
2	Баранова	Ирина	Алексеевна	1983	3	10
3	Корнилова	Ольга	Владимировна	1984	5	9
4	Воробьев	Алексей	Петрович	1983	1	10
5	Воробьев	Алексей	Иванович	1984	3	9
6	Воробьев	Олег	Григорьвич	1985	5	8
7	Скоркин	Александр	Евгеньевич	1982	1	11
8	Володина	Анна	Алексеевна	1984	3	9
9	Новоселов	Алексей	Антонович	1983	5	10
10	Алехина	Елена	Алексеевна	1984	1	9

1.11. Сохранить введенные данные, щелкнув по кнопке **Сохранить** *на панели инструментов.*

1.12. Выполнить редактирование таблицы:

– заменить фамилию *Иванникова* на фамилию *Иванова*, для чего выделить редактируемую ячейку и набрать новую фамилию;

– заменить в седьмой записи год рождения на 1985, для чего выделить редактируемую ячейку, удалить цифру 2 и ввести цифру 5.

1.13. Отсортировать ***последовательно*** значения таблицы*:*

– фамилии – *по алфавиту*;

– имя – *по алфавиту*;

– номер школы – *по убыванию*;

– год рождения – *по убыванию*;

– класс – *по возрастанию*.

Для выполнения сортировки поставить маркер на любое значение ячейки сортируемого столбца и щелкнуть по кнопке **Сортировка по возрастанию** *или* **Сортировка по убыванию** на панели инструментов.

1.14. Сохранить текущую таблицу в папке **Мои документы**, щелкнув по кнопке **Сохранить** *на панели инструментов*, а затем закрыть окно.

1.15. Открыть созданную базу данных **Список**, для чего необходимо открыть папку **Мои документы** на рабочем столе, дважды щелкнуть по имени вашей базы данных и в окне **Имя БД: база данных** выделить имя таблицы **Список** и нажать кнопку **Открыть.**

1.16. Выполнить поиск записей по образцу, для чего:

– установить текстовый курсор в любую ячейку поля **Фамилия** и щелкнуть по кнопке **Найти** *на панели инструментов*;

– в появившемся окне **Поиск и замена** набрать в поле **Образец** фамилию *Володина*, щелкнуть по вкладке **Поиск**, нажать на кнопку **Найти далее** и закрыть окно; при этом *в базе данных* **Список** будет выделена фамилия *Володина*.

1.17. Завершить работу с **Access**, для чего в меню **Файл** выбрать команду **Выход**, сохранив при этом данные.

Создание базы данных, состоящей из двух таблиц

2.1. Открыть свою базу данных, для чего:

– вызвать программу **Access** и в окне **MS Access** выделить из списка свою БД;

– включить переключатель **Открыть базу данных** и нажать кнопку **ОК.**

2.2. В окне: **Имя: база данных** выделить имя таблицы **Список** и нажать кнопку **Удалить** или клавишу **<Del>,** а затем подтвердить удаление.

2.3. Выбрать закладку **Таблицы** и нажать кнопку **Создать.**

2.4. В окне **Новая таблица** выбрать **Режим таблицы** и нажать кнопку **ОК.**

2.5. Переименовать **Поле 1**, для чего поставить курсор в любую ячейку столбца **Поле 1**, выполнить команду **Формат/Переименовать столбец,** ввести название поля **Учебная группа** и нажать клавишу **Enter.**

2.6. Переименовать **Поле 2** аналогично п. 3.5, введя название поля **Преподаватель.**

2.7. Сохранить таблицу с именем **Группы**, щелкнув по кнопке **Сохранить** *на панели инструментов*. На вопрос о создании ключевого слова ответить **НЕТ.**

2.8. Перейти в режим **Конструктор**, нажав на кнопку **Вид/Конструктор**, и просмотреть, как заданы поля. При этом:

– сделать поле **Учебная группа** ключевым, для чего поместить курсор на имя этого поля и щелкнуть на кнопке **Ключевое поле** *на панели инструментов*;

– тип данных поля **Учебная группа** сделать *числовым.*

2.9. Щелкнуть *по кнопке* **Сохранить** *на панели инструментов* и *закрыть таблицу*.

2.10. Открыть папку **Мои документы** на рабочем столе, дважды щелкнуть по имени вашей базы данных и в окне **Имя БД: база данных** выбрать закладку **Таблицы** и щелкнуть по кнопке **Создать.**

2.11. В окне **Новая таблица** выбрать **Режим таблицы** и нажать кнопку **ОК**.

2.12. Переименовать поля в соответствии с табл. 3:

Таблица 3

Старое название	Новое название
Поле 1	Код
Поле 2	Фамилия
Поле 3	Имя

Поле 4	Отчество
Поле 5	Год рождения
Поле 6	Школа
Поле 7	Класс
Поле 8	Учебная группа

2.13. Сохранить таблицу с именем **Учащиеся**, щелкнув по кнопке **Сохранить** на панели инструментов; при этом на вопрос о создании ключевого поля ответить **НЕТ.**

2.14. Перейти в режим **Конструктор** и посмотреть, как заданы поля; при этом:

– сделать поле **Код** ключевым, для чего поместить курсор на имя этого поля и нажать кнопку **Ключевое поле** *на панели инструментов*;

– тип поля **Код** – *счетчик*;

– тип полей **Фамилия, Имя, Отчество –** *текстовый***;**

– тип полей **Год рождения, Школа**, **Класс, Учебная группа** – *числовой;*

– в закладке **Подстановка** установить тип элемента управления – **Поле со списком**, *источник строк* – **Группы.**

2.15. Нажать *кнопку* **Сохранить** *на панели инструментов.*

2.16. Закрыть таблицу, щелкнув *по кнопке в правом верхнем углу*.

Контрольные вопросы

1. Для чего создается база данных?
2. Каково назначение СУБД?
3. Назовите этапы создания базы данных с помощью MS Access.
4. Какое расширение имеет файл базы данных в MS Access?
5. В чем отличие редактирования текстовых и числовых данных?
6. Какое окно в MS Access называют окном базы данных?
7. Какие элементы управления имеются в окне базы данных?
8. Каковы характерные особенности Access РБД?

Отчет должен содержать:

1. основную цель работы;
2. описание последовательности выполнения задания;
3. контрольные вопросы и ответы на них;
4. распечатки создаваемых таблиц.

Основная литература

[1-5] – согласно таблице раздела 7.

Дополнительная литература

[6-13] – согласно таблице раздела 7.

Лабораторная работа № 2. Создание таблиц базы данных в автоматизированном режиме средствами СУБД

Цель:

1. Научиться создавать **схему данных.**
2. Научиться создавать **формы** для ввода данных.
3. Закрепить навыки по работе со схемой данных.
4. Научиться создавать таблицу базы данных с помощью **Мастера таблиц.**
5. Закрепить навыки по заполнению и редактированию таблиц базы данных.
6. Закрепить навыки по добавлению и удалению записей.

Создание схемы данных

1.1. Открыть окно своей базы данных в соответствии с п. 3.1 (см. работу №1) и щелкнуть по кнопке **Схема данных** *на панели инструментов.*

1.2. В окне **Добавление таблицы** щелкнуть *по кнопке* **Таблицы**, выделить таблицу **Группы** и щелкнуть по кнопке **Добавить**. Затем выделить таблицу **Учащиеся** и вновь щелкнуть по кнопке **Добавить.**

1.3. Закрыть окно **Добавление таблицы.**

1.4. В окне **Схема данных**:

– увеличить таблицу **Учащиеся** так, чтобы были видны все ее поля;

– поставить курсор на имя поля ***Учебные группы*** в таблице **Группы** и, не отпуская кнопку мыши, перетащить ее на поле **Учебные группы** в таблице **Учащиеся**, а затем отпустить мышку.

1.5. В появившемся окне **Изменение связей:**

– включить значок **Обеспечение целостности данных;** *необходимо помнить*, что это невозможно будет сделать, если типы обоих полей будут заданы не одинаково;

– включить значок **Каскадное обновление связанных полей**; это приведет к тому, что при изменении номера группы в таблице **Группы** автоматически изменится соответствующий номер в таблице **Учащиеся;**

– включить значок **Каскадное удаление связанных записей**; это приведет к тому, что при удалении записи с номером группы в таблице **Группы** будут удалены все записи из таблицы **Учащиеся**, в которой стояли соответствующие номера групп.

1.6. Нажать кнопку **Создать**. В окне **Схема данных** появится связь 1:m между таблицами **Группы** и **Учащиеся.**

В связи *Один- ко-многим* каждой записи в таблице **Группы** соответствует несколько записей в таблице **Учащиеся**. При этом записи в таблице **Учащиеся** соответствует не более одной записи в таблице **Группы.**

1.7. Закрыть **Схему данных**, сохранив ее при этом.

1.8. В окне **Имя БД: база данных** выделить имя таблицы **Группы** и нажать кнопку **Открыть.**

1.9. В окне **Группы: Таблица** заполнить таблицу в соответствии с табл. 1:

Таблица 1

Учебная группа	**Преподаватель**
101	Верзаков С.А.

102	Белоусов А.И.
103	Масалова В.А.
104	Седлова Е.В.
105	Зачесова Т.П.

1.10. Сохранить таблицу и закрыть ее.

Создание формы для ввода данных

2.1. В окне базы данных выбрать объект **Формы** и щелкнуть по кнопке **Создать.**

2.2. В окне **Новая форма** выбрать режим **Автоформа: в столбец**, а в качестве источника данных - таблицу **Учащиеся** (использовать ниспадающее меню) и нажать кнопку **ОК.**

2.3. В окне **Учащиеся** заполнить базу данных в соответствии с табл. 2; при этом значения поля **Код** заполняются *автоматически.*

Таблица 2

Код	Фамилия	Имя	Отчество	Год рождения	Школа	Класс	Учебная группа
1	Чернова	Кристина	Ивановна	1984	1	9	101
2	Терещенко	Инна	Алексеевна	1983	3	10	103
3	Истратов	Максим	Владимирович	1984	5	9	101
4	Бондарь	Алексей	Иванович	1981	1	10	104
5	Новоселов	Алексей	Иванович	1984	3	9	105

2.4. Закрыть форму, сохранив при этом введенные данные и присвоив имя форме – **Учащиеся.**

Проверка каскадного обновления и удаления связанных полей

3.1.Открыть таблицу **Группы**, для чего в окне базы данных выделить таблицу **Группы** и нажать кнопку **Открыть.**

3.2. Исправить учебные группы на 201,202,203,204,205 соответственно, сохранить и закрыть таблицу **Группы.**

3.3. Открыть таблицу **Учащиеся** аналогично п. 3.1 и убедиться, что значение групп изменилось.

3.4. Закрыть таблицу **Учащиеся.**

3.5. Открыть таблицу **Группы** (см. п.3.1).

3.6. Удалить первую запись, ***для чего выделить всю строку*** и нажать клавишу **<Del>.**

3.7. Согласиться с проверочным вопросом, ответив **ДА.**

3.8. Закрыть таблицу **Группы** и открыть таблицу **Учащиеся**. Убедитесь, что ***запись*** *с номером 201 исчезла***.**

3.9. Закрыть таблицу **Учащиеся.**

3.10. .Завершить работу с **Access**, для чего выполнить команду **Файл/ Выход.**

Создание таблицы базы данных с помощью Мастера таблиц

4.1. Открыть окно своей базы данных в соответствии с п. 3.1 (см. работу №1), выбрать объект **Таблицы** и щелкнуть по кнопке **Создать.**

4.2. В окне **Новая таблица** выбрать вариант **Мастер таблиц** и щелкнуть по кнопке **ОК**.

4.3. В появившемся окне **Создание таблиц** выбрать:

- в поле **Образцы таблиц –** поле **Студенты**;
- в поле **Образцы полей** – поля **Код Студента**, **Адрес**, **Номер Телефона**, щелкая после каждого выбора по кнопке >. Эти поля попадут в **Поля новой таблицы.**

4.4. Щелкнуть по кнопке **Далее.**

4.5. Задать имя таблицы **Личные данные**, оставить автоматический выбор ключа и щелкнуть по кнопке **Далее.**

*Примечание***:** Access проверит связи данной таблицы с другими таблицами. Так как связи этой таблицы еще не устанавливались, то они не будут найдены автоматически. В этот момент можно установить новые связи, но **пока этого делать не надо.**

4.6. Щелкнуть по кнопке **Далее.**

4.7. Включить переключатель **Ввести данные непосредственно в таблицу** и щелкнуть по кнопке **Готово**. При этом откроется пустая таблица, у которой есть поля, но отсутствуют записи.

4.8. Добавить в таблицу **Личные данные** еще три поля *Word, Excel* и *Access,* в которых будут находиться *семестровые оценки* по этим предметам, для чего:

- выберите режим **Конструктор** на *панели инструментов*;
- добавьте в конец списка полей три поля с именами *Word, Excel* и *Access* и типом данных – *числовой.*

4.9. Щелкнуть по кнопке **Сохранить** на *панели инструментов.*

4.10. Перейти в режим таблицы, щелкнув по кнопке **Режим таблицы** на панели инструментов.

4.11. Закрыть таблицу, предварительно сохранив ее. В результате получится три таблицы, две из которых связаны, а третья нет.

4.12. В окне базы данных выбрать объект **Формы** и щелкнуть по кнопке **Создать.**

4.13. В диалогом окне **Новая форма** выбрать режим **Мастер форм.**

При этом в левой части диалогового окна появится описание **Мастера**. В нижнем поле *имя таблицы* или *запроса* в качестве источника данных не указывают, т.к. источник данных для формы следует указывать в диалоговом окне **Мастера.**

4.14. Щелкнуть по кнопке **ОК**. Появится окно **Создание форм.**

4.15. В открывшемся окне:

- выбрать в поле **Таблицы/запросы** таблицу **Личные данные**;
- выбрать в поле **Доступные поля** все поля из таблицы **Личные данные** и щелкнуть по кнопке >>, которая переносит все поля из списка;
- щелкнуть по кнопке **Далее.**

4.16. Выбрать внешний вид формы **В один столбец** и щелкнуть по кнопке **Далее.**

4.17. Выбрать стиль *Камень* и щелкнуть по кнопке **Далее.**

4.18. Задать имя формы **Общая форма** и щелкнуть по кнопке **Готово**. В результате получится форма, в которой можно менять существующие данные и вводить новые значения.

4.19. Заполнить таблицу данными (число записей должно быть не менее 10).

Примечание: Поле **Код Студента** заполняется *автоматически.*

4.20. Закрыть форму, предварительно сохранив ее.
4.21. Перейти на объект **Таблицы.**
4.22. Открыть таблицу **Личные данные** и убедиться, что в них появились данные.
4.23. Закрыть таблицу.

Ввод нового поля с типом данных OLE

5.1. Открыть в окне базы данных таблицу **Учащиеся** и перейти в режим **Конструктора.**
5.2. Добавить поле **Портрет** (ниже поля **Учебная группа**) с типом данных – **Поле объекта OLE**, оставив общие свойства поля по умолчанию, и щелкнуть по кнопке **Сохранить** *на панели инструментов*.
5.3. Перейти в режим таблицы, щелкнув по кнопке **Вид** *на панели инструментов.*
5.3. Щелкнуть мышкой по клетке, где должно быть *значение* поля **Портрет.**
5.4. Выполнить команду **Вставка/Объект/ Точечный рисунок/ ОК.**
5.5. Нарисовать портрет. Вернуться в таблицу, для чего необходимо в окне **Учащиеся: таблица** щелкнуть в правом верхнем углу рисунка. Рисунок будет обозначен словами. Чтобы увидеть портрет, необходимо дважды щелкнуть мышкой *по названию рисунка*.
5.6. Закрыть таблицу, предварительно сохранив ее.
5.7. Закрыть базу данных.

Контрольные вопросы

1. Что понимают под **Схемой данных**?
2. Зачем используют каскадное связывание полей и записей?
3. Как удаляется связь в схеме данных?
4. Каковы особенности создания таблиц с помощью **Мастера таблиц**?
5. Какие структуры данных используются в РБД?
6. Создать форму для ввода данных в режиме **Автоформа: ленточная**.
7. Нарисовать два портрета из таблицы **Учащиеся**.
8. Какие выделяют этапы создания БД Access?

Отчет к лабораторной работе

Отчет должен содержать:

- основную ***цель*** работы;
- описание ***последовательности выполнения*** задания;
- контрольные ***вопросы*** и ***ответы*** на них;
- распечатки ***создаваемых таблиц*** *и портретов.*

Основная литература

[1-5] – согласно таблице раздела 7.

Дополнительная литература

[6-13] – согласно таблице раздела 7.

Лабораторная работа № 3. Создание и использование запросов

Часть 1

Цель:

1. Закрепить навыки по редактированию таблиц.
2. Познакомиться с основными видами запросов.
3. Научиться создавать запросы на выборку различными способами.
4. Научиться создавать запросы на обновление данных.

Общие понятия

Запросы – это мощное средство обработки данных, хранимых в таблицах Access. С помощью запросов можно просматривать, анализировать и изменять данные из нескольких таблиц. Они также используются в качестве источника данных для форм и отчетов. Запросы позволяют *вычислять* итоговые значения и *выводить* их в компактном формате, подобном формату электронной таблицы, а также *выполнять* вычисления над группами записей.

Запросы можно создавать в режиме **Конструктор** и с помощью **Мастера таблиц.** Но лучше всего запросы готовить вручную с помощью **Конструктора.**

Создание запроса в режиме **Конструктор** открывает специальный бланк, называемый *бланком запроса по образцу*. Он состоит из двух областей. В верхней области отображается структура таблиц, к которым запрос адресован, а нижняя область разбита на столбцы – по одному столбцу на каждое поле будущей результирующей таблицы.

В нижней части бланка имеются *специальные* строки:

- строка **Сортировка**, при щелчке по которой открывается кнопка раскрывающегося списка, где можно выбрать метод сортировки: *по возрастанию или по убыванию;*
- строка **Условие отбора;** для каждого поля этой строки можно задавать *индивидуальное условие.*

Создание запроса на выборку

Выборка предполагает отбор записей, удовлетворяющих набору условий, из одной или нескольких таблиц. Запрос на выборку всегда возвращает таблицу, поэтому результат работы одного запроса можно использовать как исходную таблицу для другого запроса.

1. Открыть окно своей базы данных в соответствии с п.3.1 (см. работу №1). В базе данных должно быть три таблицы: **Учащиеся, Группы, Личные данные.**
2. Щелкнуть по кнопке **Схема данных** *на панели инструментов*.
3. Добавить в *схему данных* таблицу **Учащиеся**, щелкнув по кнопке **Добавить таблицу** *на панели инструментов*.
4. В появившемся окне **Добавление таблицы** выделить таблицу **Личные данные** и щелкнуть по кнопке **Добавить**, а затем – по кнопке **Закрыть.**
5. Поставить курсор на имя поля **Код студента** в таблице **Личные данные** и, не отпуская кнопку мыши, перетащить ее на поле **Код** в таблице **Учащиеся**, а затем отпустить мышку.
6. Щелкнуть по кнопке **Создать** и закрыть схему данных, сохранив ее.
7. Открыть окно свой базы данных, выбрать объект **Запросы** и щелкнуть по кнопке **Создать.**
8. В окне **Новый запрос** выбрать режим **Конструктор** и нажать кнопку **ОК.**
9. Добавить таблицы **Личные данные** и **Учащиеся**, выбирая их и щелкая по кнопке **Добавить.**

10. Закончить выбор, щелкнув по кнопке **Закрыть**. При этом появляется возможность выбора полей из разных таблиц.
11. Выбрать поля *Фамилия, Имя и Отчество* из таблицы **Учащиеся** и *Номер Телефона* – из таблицы **Личные данные**. Для этого достаточно сделать двойной щелчок мышкой по имени поля или перетащить мышкой название поля в клетки запроса.

12 Сохранить запрос, щелкнув по кнопке **Сохранить** *на панели инструментов*, ввести имя запроса **Номера телефонов** и щелкнуть по кнопке **ОК**.

12. Щелкнуть по кнопке **Запуск** на панели инструментов. В результате получается новая таблица с другим набором полей.

Задание 1. Составить запрос на адреса студенток, имя которых, *например*, «Анна» (задать самостоятельно). Запрос сохранить под именем заданной вами студентки.

Создание запроса с использованием логических операций в условии отбора

1. Повторить пункты 7,8,9 (стр. 20).

Примечание: Так как таблицы связаны, то между ними на экране появится линия связи. Если ее нет, то таблицы необходимо связать.

2. Закончить выбор, щелкнув по кнопке **Закрыть**. В результате появляется возможность выбора полей из разных таблиц.
3. Выбрать поля **Фамилия**, **Имя** и **Отчество** из таблицы **Учащиеся** и поля **Word, Excel, Access** – из таблицы **Личные данные**.
4. В строке *Условие отбора* под полями **Word**, **Excel** и **Access** поставить **4 Or 5**.
5. Щелкнуть по кнопке **Запуск** на *панели инструментов* для представления запроса.
6. Сохранить запрос с именем **Успеваемость 1**. Теперь в меню базы данных в окне **Запросы** будет показано **три** запроса.

Задание 2. Составить запрос на учащихся групп 201 и 203, которые имеют оценку по курсу «*Word*» и «*Excel*» **4** или **5**. Сохранить запрос с именем **Успеваемость 2.**

Создание вычисляемого поля

1. Повторить пункты 1 и 2 раздела 3.
2. Выбрать поля **Фамилия** и **Имя** из таблицы **Учащиеся** и поля **Word** и **Excel** – из таблицы **Личные данные.**
3. Поставить курсор на клетку правее **Excel** (на линии **Поле**).
4. Щелкнуть по кнопке **Построить** на панели инструментов.
5. В появившемся окне **Построитель выражений** напечатать вручную выражение **Среднее: ([Word]+[Excel])/2** и щелкнуть по кнопке **ОК**.
6. Это выражение подставится в *новое поле*. Нажать клавишу [Enter].
7. Сохранить запрос с именем **Среднее.**
8. Щелкнуть по кнопке **Запуск** на *панели инструментов* для представления запроса. Новое поле будет иметь имя **Среднее**.
9. Закрыть запрос.

Задание 3. Создать новую таблицу с названием **Новая группа**, с текстовыми полями **Фамилия**, **Имя**, **Отчество** и числовым полем **Учебная группа.** Для этого:

- задать ключевое поле **Код;**
- заполнить значениями**:** *Сидорова Анна Ивановна, Петрова Инна Сергеевна, Сергеева Ирина Петровна, Куликова Ольга Дмитриевна***.** Номер группы *101*;
- закрыть таблицу, предварительно сохранив ее.

Создание запроса на обновление

Обновление предполагает добавление, удаление или изменение уже существующих в таблице записей.

В нашем случае такая задача возникла в связи с тем, что школа №3 аккредитовалась как лицей, а школа №5 – как гимназия, одни ученики перешли в следующий класс, некоторые из них окончили школу, пришли новые.

Примечание: Для решения поставленной задачи необходимо исправить значения полей в таблице **Учащиеся**, а именно: номер школы **3** заменить на слово **Лицей**, а номер школы **5** – на слово **Гимназия**. Но при существующей структуре данных сделать это невозможно, т.к. поле **Школа** объявлено *числовым*, а мы пытаемся заменить его на *текстовое*. Поэтому необходимо предварительно сменить тип поля **Школа** *числовой* на *текстовый*, для чего:

7. Перейти на объект Таблица в своей базе данных и выделить таблицу Учащиеся.
8. Щелкнуть по кнопке Конструктор и исправить тип поля Школа на текстовый.
9. Сохранить таблицу и затем закрыть ее.
10. В окне своей базы данных выбрать объект Запрос.
11. Щелкнуть мышкой по кнопке Создать.
12. В появившемся диалоговом окне выбрать режим Конструктор и щелкнуть по кнопке ОК.
13. Добавить таблицу Учащиеся, выбрав ее из списка и щелкнув по кнопке Добавить.
14. Закончить выбор, щелкнув по кнопке Закрыть.
15. Выбрать поле Школа из таблицы Учащиеся, щелкнув по нему мышкой два раза.
16. Щелкнуть по стрелке с кнопкой Тип запроса на панели инструментов и выбрать команду Обновление.
17. Ввести условие замены: в поле Школа заменить все цифры 3 на слово Лицей. Условие отбора представлено на рис. 1.

Поле:	Школа
Имя таблицы:	Учащиеся
Обновление:	Лицей
Условие отбора:	3
или:	

Рис. 1 Условие отбора для создания запроса

1. Щелкнуть по кнопке **Запуск** на *панели инструментов***.**
2. Подтвердить обновление записей.

Примечание: Если сейчас перейдете в режим **таблицы**, то не увидите ни одной записи, т.к. цифры **3** заменились на слово **Лицей**, а *условие* – на выбор школы **3** не дает ни одной записи. Удалив в строке *Условие отбора* цифру **3**, можно увидеть в режиме **таблицы** результат замены.

3. Закрыть запрос, сохранив его с именем **Лицей.**
4. Выбрать объект **Таблицы.**
5. Открыть таблицу **Учащиеся** и просмотреть результат применения запроса.

Задание 4. Создать запрос **Гимназия**, меняющий значения поля школа **5** на слово **Гимназия** (по аналогии с запросом **Лицей**).

Отчет к работе составляется после выполнения второй части.

Часть 2

Создание и использование запросов

Цель:

1. Продолжить знакомство с основными видами запросов.
2. Научиться формировать запросы на создание таблицы.
3. Научиться создавать перекрестные запросы.

Построение запроса на добавление

1. Выбрать объект **Запросы** в своей базе данных и щелкнуть мышкой по кнопке **Создать.**
2. В появившемся окне **Новый запрос** выбрать режим **Конструктор** и щелкнуть мышкой по кнопке **ОК.**
3. Добавить таблицу **Новая группа,** выбрав ее и щелкнув по кнопке **Добавить.**
4. Закончить выбор, щелкнув по кнопке **Закрыть.**
5. Выбрать те поля, которые нужно добавить, а именно: *Фамилия, Имя, Отчество, Учебная группа.* (Если у какого-либо поля поставить условие отбора, то добавятся только записи, удовлетворяющие этому условию).
6. Щелкнуть по стрелке рядом с кнопкой **Тип запроса** на *панели инструментов* и выполнить команду **Добавление.**
7. В появившемся окне **Добавление** выбрать имя таблицы **Учащиеся**, *в которую будут добавляться данные*, и щелкнуть по кнопке **ОК.**
8. Щелкнуть по кнопке **Запуск** на панели инструментов для выполнения запроса, а затем *подтвердить* выполнение запроса.

Примечание: Если появится ошибка в выполнении данного запроса, то нужно открыть схему данных, щелкнув дважды по связи **Учащиеся – Личные данные**, удалить условие **Каскадное удаление связанных полей** и **Каскадное обновление связанных полей**. Вероятно, вы нечаянно их включили, хотя *в работе 3 этого не требовалось.*

9. Закрыть запрос, сохранив его с именем **Добавление.**
10. Перейти на объект **Таблицы**, открыть таблицу **Учащиеся** и убедиться, что данные записи были добавлены.

Построение запроса на удаление

Требуется удалить записи тех учащихся, которые окончили школу.

Примечание: Для решения поставленной задачи **предварительно** необходимо разорвать связь *Учащиеся – Личные данные.*

1. Повторить п.1 и 2., указанные на стр. 24.
2. Добавить таблицу **Учащиеся**, выбрав ее из списка и щелкнув по кнопке **Добавить.**
3. Закончить выбор, щелкнув по кнопке **Закрыть.**
4. Щелкнуть по стрелке рядом с кнопкой **Тип запроса** на *панели инструментов* и выполнить команду **Удаление….**
5. Выбрать поле **Класс** из таблицы **Учащиеся**, щелкнув по нему мышкой два раза.
6. Ввести условие отбора **11** (рис.2).

Поле:	Класс
Имя таблицы:	Учащиеся
Удаление:	Условие
Условие отбора:	11

Рис.2 Условие отбора для создания запроса

7. Щелкнуть по кнопке **Запуск** для выполнения запроса. *Появится сообщение, что процесс удаления будет необратим.* Подтвердить удаление записей.
8. Закрыть запрос, сохранив его с именем **Удаление.**
9. Перейти на объект **Таблицы**, открыть таблицу **Учащиеся** и убедиться, что данные записи удалены.

Задание5. Создать запросы на обновление с именами **Десятый** и **Девятый**, которые меняют класс на единицу больше (10 на 11 и 9 на 10). Заполнить недостающие данные для добавленных записей, указав для них класс **9.**

Построение запроса на создание таблицы

Такая задача возникает тогда, когда необходимо из старых таблиц составить новые таблицы с другим набором полей.

Примечание: Запрос на создание таблицы отличается от запроса на выборку тем, что *результат запроса сохраняется как таблица базы данных*, после чего эта таблица может быть включена в состав других таблиц, на которых может строиться запрос.

Пусть требуется создать таблицы успеваемости для учащихся разных групп. Для этого необходимо:

1. Повторить п.1 и 2. раздела 6.
2. Добавить таблицы **Учащиеся** и **Личные данные**, выбрав их и щелкнув по кнопке **Добавить.**
3. Закончить выбор, щелкнув по кнопке **Закрыть.**
4. Щелкнуть по стрелке рядом с кнопкой **Тип запроса** на *панели инструментов и* выполнить команду **Создание таблицы ….**
5. Напечатать имя таблицы **Успеваемость** и щелкнуть по кнопке **ОК**
6. Выбрать поля **Фамилия, Имя, Отчество** и **Учебная группа** из таблицы **Учащиеся** и поля **Word, Excel** и **Access –** из таблицы **Личные данные.**
7. Щелкнуть по кнопке **Запуск** для выполнения запроса на *панели инструментов.*
8. Подтвердить выполнение запроса.
9. Закрыть запрос, сохранив его с именем **Новая таблица.**
10. Перейти на объект **Таблицы**, открыть таблицу **Успеваемость** и убедиться, что записи добавлены. Но при этом добавится только 10 записей, т.к. в таблицу **Личные данные** дополнительные записи не вошли.

Создание перекрестного запроса

Перекрестный запрос строится *на основе одной таблицы*. Результатом перекрестного запроса является таблица статистических расчетов (**sum** – сумма, **count** – количество записей, **avg** –среднее значение, **max, min** …), выполненных по данным *из одного поля* таблицы. Например, требуется подсчитать для экзаменационной ведомости, сколько в группе, занимающейся изучением **Word**, получено *«троек», «четверок» и «пятерок».* Для этого необходимо:

1. Выбрать объект **Запрос** и щелкнуть по кнопке **Создать**.
2. В появившемся диалоговом окне выбрать **Перекрестный запрос** и щелкнуть по кнопке **ОК**.
3. В окне **Создание перекрестных запросов** выделить таблицу **Успеваемость** и щелкнуть по кнопке **Далее**.
4. Выбрать *поле*, значения которого будут использоваться в качестве *заголовков строк*; в нашем примере это **Учебная группа.** Щелкнуть по кнопке **Далее**.
5. Выбрать *поле*, значение которого будет использоваться в качестве столбцов; в нашем примере это поле **Word**. Щелкнуть по кнопке **Далее**.
6. Выбрать функцию, по которой будут вычисляться значения ячеек на пересечении столбцов и строк; в нашем примере функция **Count** – *количество*. Щелкнуть по кнопке **Далее.**
7. Задать имя запроса **Word** и щелкнуть по кнопке **Готово.**
8. Закрыть таблицу запроса.

Задание 6. Составить запросы для оценок, полученных группой по изучению Excel и Access.

Создание запроса с параметрами

Запрос с параметром – это запрос, при выполнении которого в диалоговом окне *пользователю* выдается *приглашение* ввести данные, на основе которых выполнится запрос.

Например, требуется получить информацию о номере телефона студента, т. е. требуется создать запрос, в котором должна выводиться фамилия и номер телефона студента. Для этого необходимо:

1. Создать новый запрос на выборку.
2. Открыть его в режиме **Конструктор.**
3. В строку **Условие отбора** ввести ***в квадратных скобках***:
   - для поля **Фамилия** текст *Фамилия студента*;
   - для поля **Номер телефона** текст *Номер телефона студента.*
4. Запустить запрос. При выполнении запроса Access выведет диалоговые окна, в которые пользователь сможет ввести нужные значения параметров.

Завершить работу с СУБД MS Access, для чего выполнить команду **Файл – Выход**.

Примечание: Если производили редактирование в базе данных, то на вопрос о сохранении изменений ответьте утвердительно.

Контрольные вопросы

1. Каково назначение запросов?
2. Каковы особенности результирующего набора данных запроса?
3. В чем достоинства запросов?
4. Для чего предназначены запросы на выборку?
5. В чем особенность запроса на выборку?
6. Для чего предназначены запросы с использованием логических операций в условии отбора?
7. Какие условия необходимо соблюдать при создании запроса на обновление?
8. Какова особенность запроса на создание таблицы?
9. Для чего предназначены перекрестные запросы?
10. Каковы правила конструирования условий отбора?

Отчет к лабораторной работе

Отчет должен содержать:

- основную цель работы;
- описание последовательности выполнения задания;
- контрольные вопросы и ответы на них;
- распечатки *создаваемых запросов и результирующих таблиц.*
- описание последовательности выполнения заданий.

Основная литература

[1-5] – согласно таблице раздела 7.

Дополнительная литература

[6-13] – согласно таблице раздела 7.

Лабораторная работа № 4. Создание экранных форм и отчетов

Цель

Получение практических навыков и умений в создании экранных форма, отчетов.

Общие сведения

Экранная форма имеет *три* основных раздела: *область заголовка*, *область данных* и *область примечания*. Линии, разделяющие разделы, перетаскиваются по вертикали с помощью мыши, что позволяет изменять размеры разделов так, как требуется. *Элементы управления* для разработчика *представлены* на **Панели инструментов**, которая открывается командой **Вид/Панель элементов** *только* в режиме **Конструктор**. *Выбор* элемента управления выполняется одним щелчком на его значке в **Панели элементов**, после чего следующим щелчком в *поле формы* вставляется его присоединенная надпись. *Редактированием* свойства элемента управления можно дать элементу управления более содержательную подпись через контекстное меню.

Отчеты предназначены для того, чтобы *выбирать* данные из нескольких таблиц, *производить* над ними вычисления, *подводить* итоги и *выводить* их на экран или печать. Они имеют широкие возможности для группировки, а так же вычисления промежуточных и общих итогов для больших наборов данных. Отчеты, как и формы, имеют много общих конструктивных черт. Как и формы, отчеты также имеют заголовки и области данных, а так же есть возможность создавать подчиненные отчеты и вставлять их в другие отчеты.

Создание однотабличной экранной формы

1.1. Открыть окно *своей базы данных* и выделить форму **Учащиеся**.

1.2. Выбрать режим **Конструктор** на панели инструментов.

1.3. В открывшемся окне **Учащиеся: форма** *ввести* текст заголовка, для чего:

- расширить область заголовка формы;
- создать графический элемент **Надпись (Аа);**
- выбрать нужный шрифт (например, *Times New Roman*) и размер (например, *14*);
- растянуть рамку текста до нужного размера;
- ввести текст **Учащиеся-1**;
- щелкнуть мышкой вне рамки элемента.

1.4. Отформатировать элемент **Надпись**, для чего:

- выделить щелчком мыши элемент внутри рамки;
- перевести на текст курсор;
- изменить границы элемента, если необходимо;
- щелкнуть мышкой вне рамки элемента.

1.5. Сохранить экранную форму, для чего нажать кнопку сохранить *на панели инструментов*.

Редактирование многотабличной экранной формы:

- в окне своей базы данных выбрать объект Формы и выделить форму Преподаватели -1;
- выбрать режим Конструктор;
- в открывшемся окне Преподаватели-1: форма осуществить доработку формы аналогично пунктам 1.3, 1.4 и 1.5, т.е. ввести текст заголовка Данные по преподавателям, отформатировать элемент Надпись и сохранить форму.
- в окне своей базы данных выбрать объект Формы и выделить форму Учащиеся-1;
- выбрать режим Конструктор;
- в открывшемся окне Учащиеся-1: форма осуществить доработку формы аналогично пунктам 1.3, 1.4 и 1.5, т.е. ввести текст заголовка Данные по студентам, отформатировать элемент Надпись и сохранить форму.

Примечание:

Изменение размеров любого элемента осуществляется перемещением границ его рамки.

Создание отчетов

По своим свойствам и структуре *отчеты* во многом *похожи* на *формы*, но *предназначены* для вывода на печатающее устройство. Они *отличаются* тем, что в них приняты специальные меры для *группирования* выводимых данных и для *вывода* специальных элементов оформления, характерных для печатных документов (верхний и нижний колонтитулы, номера страниц, служебная информация о времени создания отчета и т.п.).

Перед началом конструирования отчета необходимо *спроектировать* его ***макет***. При этом определяют состав и содержание разделов отчета, размещение в нем значений, выводимых из полей таблиц базы данных, а также *определяют:*

- поля, по которым нужно группировать данные;
- заголовки, примечания и вычисляемые итоговые значения для каждого уровня группировки;
- порядок вывода данных в отчете, а также оформляют заголовки и подписи атрибутов отчета.

Создание и *изменение макета отчета* осуществляют в окне К**онструктор отчетов.** При этом в окне первоначально отображаются *пустые* разделы отчета:

- ***заголовок отчета*** – все, что находится в этой области, выводится только один раз в начале отчета;
- ***верхний колонтитул*** - все, что находится в этой области, выводится в верхней части каждой страницы;
- ***область данных*** – содержит собственно записи;
- ***нижний колонтитул*** - все, что находится в этой области, выводится в нижней части каждой страницы;
- ***примечание отчета*** - все, что находится в этой области, выводится только один раз в конце отчета.

Наличие этих разделов, а также их *удаление* или *включение* определяют командами меню **Вид/Колонтитулы** и **Вид/Заголовок/Примечание отчета**. Для этих же целей можно использовать соответствующие кнопки панели инструментов конструктора отчетов.

Области *примечаний* и *заголовок* отличаются от остальной части окна – *области данных* только тем, что не двигаются относительно окна при перемещении по данным в случае ленточной или многострочной формы, что означает возможность использования в этих областях любых элементов управления и ввода/вывода в форму.

При *создании* ***отчета*** его разделы *заполняют* элементами в соответствии *с разработанным* ***макетом*** отчета. В заголовок помещают текст из *шапки макета* отчета. В верхний и нижний колонтитул обычно помещают заголовки, номера страниц и даты. В области данных размещают поля таблиц базы данных или запросов. В примечании могут быть помещены выражения для подведения итогов по группе.

Поля *с неповторяющимися* значениями размещают в **Области данных**, которой можно придать вид табличной части отчета. Поля *с повторяющимися* значениями, по которым производят группировки записей, как правило, размещают в *заголовке* группы.

Средством автоматизированного создания отчетов является **Мастер отчетов**. Он работает в несколько этапов. При его работе выполняется выбор базовых таблиц или запросов, на которых отчет базируется, выбор полей, отображаемых в отчете, выбор полей группировки, выбор полей и методов сортировки, выбор формы печатного текста и стиля оформления.

2.1. В окне своей базы данных открыть объект **Отчеты** и щелкнуть по кнопке **Создать.**

2.2. В появившемся диалоговом окне **Новый отчет** выбрать режим **Автоотчет: в столбец,** а в качестве источника данных запрос **Успеваемость 2.**

2.3. Щелкнуть по кнопке ОК; появится страница просмотра отчета.

2.4. Сохранить *отчет* с именем **Успеваемость2** и закрыть отчет.

Примечание: Этот отчет составлен на основании запроса по *заданию 2.* При изменении запроса **Успеваемость 2** изменится и отчет. Это дает возможность, например, распечатать группу только для одного студента. Для этого:

- открыть объект **Запросы** в окне своей базы данных и выделить запрос **Успеваемость 2**;
- выбрать режим **Конструктор**;
- ввести условие отбора *фамилии*: например **Баранов** (выбирает студент в соответствии с его списком);
- выполнить запрос, щелкнув по кнопке **Запуск** на *панели инструментов;*
- сохранить запрос с именем **Студент** и закрыть его;
- перейти на объект **Отчеты** и щелкнуть по кнопке **Создать;**

- повторить п. 2.2 и 2.3, выбрав в качестве источника данных запрос **Студент;**
- **у**бедиться, что в списке вывода будут находиться данные только для одного студента;
- сохранить отчет с именем **Студент** и закрыть отчет.

Задание. Создать ленточный автоотчет на основании запроса **Номера телефонов.**

2.5. Завершить работу с СУБД **MS Access**.

Контрольные вопросы

1. Для чего предназначена экранная форма?
2. Для чего создают многотабличные экранные формы?
3. Какие данные располагают в области данных экранной формы?
4. Для чего предназначены отчеты?
5. Для чего проектируют макет отчета?
6. В чем особенность построения отчета?

Отчет к лабораторной работе

Отчет должен содержать:

- основную цель работы;
- описание последовательности выполнения задания;
- контрольные вопросы и ответы на них;
- распечатки создаваемых экранных форм и отчетов.
- описание последовательности предложенного к выполнению **задания с распечатками запроса и отчета.**

Основная литература

[1-5] – согласно таблице раздела 7.

Дополнительная литература

[6-13] – согласно таблице раздела 7.

Лабораторная работа № 5. Определение сущностей и атрибутов предметной области

Цель:

1. Научиться выделять сущности, атрибуты заданной предметной области.
2. Определять связи между сущностями.

Общие понятия.

Сущности и атрибуты концептуальной модели

Концептуальное (инфологическое) проектирование — построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных.

Термины «семантическая модель», «концептуальная модель» и «инфологическая модель» являются синонимами. Кроме того, в этом контексте равноправно могут использоваться слова «модель базы данных» и «модель предметной области» (например, «концептуальная модель базы данных» и «концептуальная модель предметной области»), поскольку такая модель является как образом реальности, так и образом проектируемой базы данных для этой реальности.

Концептуа́льная моде́ль (англ. conceptual model) — это модель, представленная множеством понятий и связей между ними, определяющих смысловую структуру рассматриваемой предметной области или ее конкретного объекта.

Модели «сущность-связь» (англ. “Entity-Relationship model”), или ER-модель, предложенная П. Ченом в 1976 г., является наиболее известным представителем класса семантических (концептуальных, инфологических) моделей предметной области. ER-модель обычно представляется в графической форме, с использованием оригинальной нотации П. Чена, называемой ER-диаграмма.

Основные преимущества ER-моделей:

- наглядность;
- модели позволяют проектировать базы данных с большим количеством объектов и атрибутов;
- ER-модели реализованы во многих системах автоматизированного проектирования баз данных (например, ERWin).

Основные элементы ER-моделей:

- объекты (сущности);
- атрибуты объектов;
- связи между объектами.

Сущность — объект предметной области, имеющий атрибуты.

Связь между сущностями характеризуется:

- типом связи (1:1, 1:N, N:M);
- классом принадлежности.

Класс может быть обязательным и необязательным. Если каждый экземпляр сущности участвует в связи, то класс принадлежности — обязательный, иначе — необязательный.

В качестве примера, рассмотрим набор объектов, характеризующий сотрудников некоторой фабрики. В качестве атрибутов можно указать следующее:

Название атрибута	Тип данных	Домен
Фамилия	Литерный	Сочетание символов-букв
Имя	Литерный	Сочетание символов-букв
Отчество	Литерный	Сочетание символов-букв
Номер отдела	Числовой	Любое положительное целое число
Должность	Литерный	Сочетание символов-букв
Дата рождения	Тип дата	Допустимые значения при описании даты
Стаж	Числовой	Любое положительное целое число
Характеристика	Текст	Любой текст
Табельный номер	Числовой	Любое положительное целое число

Задания

5.1. Опишите набор сущностей, задающий совокупность студентов заданного факультета. Укажите перечень атрибутов с указанием типа данных, отвечающих каждому из них. Для каждого из атрибутов указать домен.

5.2. Опишите набор сущностей, задающий учебный план заданного факультета. Считать, что на факультете возможна специализация по нескольким специальностям. Укажите перечень атрибутов с указанием типа данных, отвечающих каждому из них. Необходимо указать также свойство объекта, описываемое каждым их атрибутов. Для каждого из атрибутов указать домен.

5.3. Опишите набор сущностей описывающих некоторый оптовый склад торговой фирмы. Считать, что фирма получает товар от различных поставщиков. Укажите перечень атрибутов с указанием типа данных, отвечающих каждому из них. Необходимо указать также свойство объекта, описываемое каждым их атрибутов. Для каждого из атрибутов указать домен.

5.4. Опишите набор сущностей, описывающих совокупность товаров некоторого частного магазина. Считать, что магазин получает товар с различных оптовых складов и различных фирм-поставщиков. Укажите перечень атрибутов с указанием типа данных, отвечающих каждому из них. Необходимо указать также свойство объекта, описываемое каждым их атрибутов. Для каждого из атрибутов указать домен.

5.5. Опишите набор сущностей, задающий книжный фонд некоторой библиотеки. Укажите перечень атрибутов с указанием типа данных, отвечающих каждому из них. Необходимо указать также свойство объекта, описываемое каждым их атрибутов. Для каждого из атрибутов указать домен.

5.6. Опишите набор сущностей, задающий список читателей некоторой библиотеки. Укажите перечень атрибутов с указанием типа данных, отвечающих каждому из них. Необходимо указать также свойство объекта, описываемое каждым их атрибутов. Для каждого из атрибутов указать домен.

Контрольные вопросы

1. Что понимается под предметной областью?
2. Что такое концептуальное моделирование, концептуальная модель?
3. Назовите основные преимущества и элементы ER-моделей?
4. Что понимается под сущностью, атрибутами предметной области?
5. Что такое отношение, кортеж, домен?
6. Существующие типы связей между сущностями?
7. Какие классы принадлежности между сущностями существуют?
8. Определите понятия категорийная и ссылочная целостность?
9. Что такое внешний ключ отношения?

Отчет к лабораторной работе

Отчет должен содержать:

- основную цель работы;
- описание последовательности выполнения задания;
- контрольные вопросы и ответы на них;
- описание сущностей предметной области и атрибутов.

Основная литература

[1-5] – согласно таблице раздела 7.

Дополнительная литература

[6-13] – согласно таблице раздела 7.

Лабораторная работа № 6. Нормализация данных

Цель

Закрепление теоретического материала и получение практических навыков проведения нормализации отношений и приведения схемы данных к третьей нормальной форме или к НФБК.

Нормализация представляет собой процесс реорганизации таким образом, чтобы исключить повторения одних и тех же сведений и иных противоречий. Окончательная цель нормализации сводится к получению такого проекта базы данных, в которой каждый факт появляется один раз в одном месте (исключение избыточности информации), и, кроме того, исключить противоречия в хранимых данных. Целью нормализации является освободить проект от избыточности данных, аномалии обновления, аномалии удаления, аномалии ввода.

Основные проблемы, возникающие при работе с ненормализованными таблицами:

- избыточность данных;
- аномалия обновления;
- аномалия удаления;
- аномалия ввода.

Задание

Разработать в представленных ниже задачах схему данных в третьей нормальной форме.

Задания

При проектировании базы данных в представленных ниже заданиях необходимо привести соответствующую схему данных к третьей нормальной форме.

6.1 Построить базу данных, обслуживающую ведение заказов авторемонтной мастерской. Информация должна содержать сведения о клиенте(ФИО, адрес), тип работы, оплату и информацию об исполнителе (ФИО, квалификация).

6.2.

Построить базу данных, описывающую результаты сессии. Информация должна содержать номер семестра, сведения о студенте (ФИО, группа, специальность), сведения о сдаваемом предмете (название, семестр), дату сдачи экзамена, оценку и ФИО экзаменатора.

6.3.

Построить базу данных, описывающую работу библиотеки с читателем. Информация должна содержать сведения о читателе (ФИО, адрес, телефон), информацию о выданной книге (название, автор, издательство) и дату выдачи книги.

6.4.

Построить базу данных, описывающую обращение больных в поликлинику. Информация должна содержать сведения о больных (ФИО, адрес, дату рождения), враче (ФИО, специальность), дате осмотра и заключение врача.

6.5.Построить базу данных, описывающую работу с заказами некоторой оптовой базы. Информация должна содержать сведения о заказчике (Название фирмы, адрес, телефон), сведения о заказываемом товаре (Наименование, фирма изготовитель, год выпуска, стоимость единицы продукции), а также количество заказанного товара.

6.6.Построить базу данных, описывающую формирование фонда сети магазинов некоторой фирмы. Информация должна содержать сведения о магазине (название, адрес, телефон), сведения о поставщике (наименование, адрес, телефон) сведения о товаре (наименование, количество) и дату поставки.

6.7.Построить базу данных, описывающую работу с клиентами фирмы по техническому обслуживанию торгового оборудования. Информация должна собираться о мастерах, выполняющих ремонтные работы (ФИО, квалификация, телефон), о магазинах, подающих заявки на ремонт оборудования (наименование оборудования, магазин, адрес, телефон) и о выполнении заказа с указанием даты выполнения и оплате.

6.8.Построить базу данных, описывающую репертуарную политику театра. Информация собирается об актерах (ФИО, звание, дата рождения, адрес, телефон), о пьесе (авторы, название, список ролей с указанием их характеристики, то есть возраст, амплуа и так далее) и о репертуаре на следующий месяц с указанием даты спектакля.

6.9.Построить базу данных, описывающую репертуарную политику филармонии. Информация собирается об исполнителях (ФИО или название коллектива, адрес, телефон, дополнительные сведения о них), об исполняемых произведениях, концертной площадке (название, характеристика, объем), контактный телефон, дате концерта и времени его начала.

6.10.Построить базу данных, описывающую проведение чемпионата высшей лиги по футболу. Информация должна содержать сведения о клубе (название, главный тренер, место дислокации), футболистах (ФИО, дата рождения, номер игрока, специализация), места проведения матча (город, площадка) и даты проведения матча и счет.

6.11.Построить базу данных, описывающую работу страховой компании. Информация должна содержать сведения о компании (название, номер регистрации, ФИО агента, телефон связи), о видах страхования, о клиенте (ФИО, адрес, телефон), дату заключения сделки, страховую сумму и комиссионные.

6.12.Построить базу данных, описывающую деятельность ремонтной бригады ЖКХ. Информация должна содержать сведения о работниках бригады (ФИО, квалификация, специальность), сведения о заказчике (ФИО, адрес, телефон), контактный телефон ЖКХ вид ремонта и дату выполнения заказа.

6.13.Построить базу данных, описывающую работу фермерского хозяйства. Информация должна содержать сведения о наемных работниках (ФИО, адрес, дата рождения), о проводимых работах (название, оплата), дату начала и окончания работы .

6.14Построить базу данных, описывающую проведение чемпионата высших учебных заведений по баскетболу. Информация должна содержать сведения об учебном заведении (название, главный тренер, место нахождения), о членах команды (ФИО, дата рождения, номер игрока, факультет, группа), места проведения матча (город, площадка) и даты проведения матча и счет.

6.15.Построить базу данных, описывающую работу центра занятости. Информация должна содержать сведения о работодателях (Название, адрес, телефон, должность, квалификация, ставка), о потенциальных претендентах (ФИО, адрес, телефон, дата рождения, квалификация, стаж работы) и дату заключения договора о найме.

6.16.Построить базу данных, описывающую работу бригады ремонта дорожных покрытий. Информация должна содержать сведения о сотрудниках бригады (ФИО, адрес, телефон, специальность), о техническом парке (наименование, количество), о месте проведения и объеме работ, исполнителях, дате начала и окончания работы.

6.17.Построить базу данных, описывающую ведение журнала успеваемости в школе. Информация должна содержать сведения о школьнике (ФИО, день рождения, адрес, телефон, сведения об отце и матери, класс), о преподающихся дисциплинах (название, класс), дату ответа и оценку.

6.18.Построить базу данных, описывающую проведение зимней универсиады. Информация должна содержать сведения об участниках (ФИО, место жительства, город, название университета, дату рождения), список дисциплин универсиады, место проведения соревнования, дату проведения и список участников и показанные результаты.

6.19.Построить базу данных, описывающую работу фотоателье. Информация должна содержать сведения о сотрудниках фотоателье (ФИО, адрес, телефон, должность), сведения о клиенте (ФИО, адрес), дату проведения съемки и дату выполнения заказа.

Контрольные вопросы

1. Что понимается под нормализацией отношений?
2. Определите цель проведения нормализации?
3. Определите такие ситуации, как избыточность данных, аномалия обновления, аномалия удаления, аномалия ввода, которые возникают при работе с таблицами?
4. Сколько существует видов нормальных форм? Определите каждую из них.
5. Что такое НФБК?
6. Какая зависимость между атрибутами называется функциональной?
7. Какая функциональная зависимость атрибутов называется транзитивной?
8. Как определяется полная функциональная зависимость?

Основная литература

[1-5] – согласно таблице раздела 7.

Дополнительная литература

[6-13] – согласно таблице раздела 7.

Лабораторная работа № 7. Создание базы данных

При выполнении лабораторной работы преподаватель назначает каждому студенту свой вариант предметной области или обучающийся выбирает предметную область самостоятельно, но обязательно согласовывает ее с преподавателем до выполнения работы.

Цель

Закрепление теоретических знаний и практических навыков создания базы данных для конкретной предметной области с использованием конкретной СУБД.

Задание

В процессе создания базы данных необходимо провести описание предметной области, осуществить выбор структур таблиц и обоснование данного выбора, наложение условий целостности, определить ключи, внешние ключи, а также указать поля и ограничения, налагаемые на поля.

Содержание

1. Создать таблицы, описанные в предметной области (варианты предметных областей приведены далее).

2. Для каждой создаваемой таблицы:
- определить условия на значения и сообщения об ошибках некоторых полей;
- определить начальное значение для некоторых полей;
- определить ключ;
- определить внешний ключ (если он есть);
- определить (если это возможно) значения некоторых полей с помощью мастера подстановок;
- определить обязательные поля.

3. Ввести данные в таблицы. При вводе выяснить, что дает наложение условий на значения полей.

4. Определить схему базы данных, связи между таблицами и наложить условия целостности на таблицы, связанные отношением "один-ко-многим". Показать на примерах, что меняется при включении/выключении каждого из флажков "Обеспечение целостности данных" и "каскадное обновление связанных записей" и "каскадное удаление связанных записей".

Контрольные вопросы

1. Что такое ключ, внешний ключ?
2. Что такое идентификатор?
3. Что такое возможный ключ?
4. Определите 3НФ?
5. Что понимается под обеспечением целостности, каскадным обновлением?

Основная литература

[1-5] – согласно таблице раздела 7.

Дополнительная литература

[6-13] – согласно таблице раздела 7.

Лабораторная работа № 8. Создание SQL-запросов

Цель

Получение теоретических знаний и практических навыков в использовании языка SQL при создании запросов на выборку.

Задание

При выполнении данной лабораторной работы каждый выполненный пункт необходимо сохранять в виде отдельного запроса.

1. Создать простой запрос на выборку из одной таблицы. Включить несколько полей таблицы.
2. Включить в запрос все поля с помощью знака "*".
3. Запрос из нескольких связанных таблиц. Добавление и удаление таблиц из запроса.
4. Ввод данных с помощью запроса одновременно в родительскую и дочернюю таблицу.
5. Выбрать несколько полей, по которым сортируется вывод.
6. Определить условия отбора ("И" и "ИЛИ").
7. Определение условий отбора с помощью параметра запроса.
8. Создать вычислимые поля.
9. Создать отсортированный по вычислимому полю запрос из нескольких таблиц, в котором определены условия "И" и "ИЛИ".

Контрольные вопросы

1. Что определяют SQL, QBE?
2. Что понимают под DDL?
3. Определите содержание языка SQL.
4. Что понимают под простым запросом?
5. Как реализуют запрос с параметром?

Основная литература

[1-5] – согласно таблице раздела 7.

Дополнительная литература

[6-13] – согласно таблице раздела 7.

Лабораторная работа № 9. Создание группирующих и перекрестных запросов на SQL

Цель

Изучение языка SQL и получение практических навыков в создании группирующих запросов, группирующих запросов с условием, перекрестных запросов.

Содержание

При выполнении данной работы каждый выполненный пункт необходимо сохранять в виде отдельного запроса.

1. Создать итоговый запрос, содержащий несколько итоговых цифр.
2. Создать простой группирующий запрос.
3. Создать группирующий запрос с группировкой по нескольким полям.
4. Создать группирующий запрос, в котором определяются условия, причем сначала выполняются вычисления, а затем происходит отбор.
5. Создать группирующий запрос, в котором определяются условия, причем сначала происходит отбор, а затем выполняются вычисления.
6. Создать группирующий запрос, в котором есть вычислимое выражение, содержащее несколько итоговых полей.
7. Создать с помощью мастера перекрестный запрос.

Контрольные вопросы

1. Как создать группирующий запрос?
2. Как создать группирующий запрос с группировкой по полям?
3. Что такое перекрестный запрос?
4. Какими способами можно создать перекрестный запрос?

Основная литература

[1-5] – согласно таблице раздела 7.

Дополнительная литература

[6-13] – согласно таблице раздела 7.

Лабораторная работа № 10. Создание запросов действия и перекрестных запросов на SQL

Цель

Изучение языка SQL и получение практических знаний и навыков использования запросов действия и запросов объединения с применением SQL.

Содержание

1. Создать с помощью запроса новую таблицу.
2. Удалить записи с помощью запроса удаления.
3. Добавить записи с помощью запроса добавления.
4. Обновить записи с помощью запросов на обновление.

Контрольные вопросы

1. Как создать запрос действия?
2. Что такое перекрестный запрос?
3. Какими способами можно создать перекрестный запрос?
4. Как создать новую таблицу с помощью запроса?
5. Как добавить, обновить и удалить записи в таблице с помощью запроса?

Основная литература

[1-5] – согласно таблице раздела 7.

Дополнительная литература

[6-13] – согласно таблице раздела 7.

Лабораторная работа № 11. Создание клиентского приложения

Цель Получение практических навыков в создании приложения пользователя и разработки руководства пользователя.

Содержание

1. Создать формы для ввода каждой из таблиц-справочников.
2. Создать сложную форму для таблиц, связанных отношением «1 ко многим».
3. Создать кнопочную форму, которая бы предоставляла доступ ко всем созданным формам и запросам.
4. Поместить в созданные формы кнопки навигации по записям и работы с формой (закрыть, напечатать, выйти из приложения и т.д.).
5. Создать макрос для автоматической загрузки кнопочной формы при открытии базы данных.
6. Разработать руководство пользователя.

Отчет к лабораторной работе

Отчет должен содержать:

- основную цель работы;
- описание последовательности выполнения задания;

распечатки схем данных и создаваемых экранных форм и отчетов;

- распечатку разработанного руководства пользователя.

Варианты предметной области

Вариант 1. Страховая компания.

Договоры (НомерДоговора, Дата заключения, Страховая сумма, Тарифная ставка, Код филиала, Код вида страхования)

Вид страхования (КодВида, Наименование, Комиссионное вознаграждение)

Филиал (КодФилиала, Наименование филиала, Адрес, Телефон)

Вариант 2. Страховая компания.

Договоры (НомерДоговора, Дата заключения, Страховая сумма, Тарифная ставка, Код филиала, Код вида страхования)

Вид страхования (КодВида, Наименование, Комиссионное вознаграждение)

Выплаты (КодВыплаты, Номер договора, Дата выплаты, СуммаВыплаты)

Вариант 3. Страховая компания.

Выплаты (НомерДоговора, Дата заключения, Страховая сумма, СуммаВыплаты, Код филиала, Код вида страхования)

ВидСтрахования (КодВида, Наименование, Комиссионное вознаграждение)

Филиал (КодФилиала, Наименование филиала, Адрес, Телефон).

Вариант 4. Реализация готовой продукции.

Товары(КодТовара, Наименование, ОптоваяЦена, РозничнаяЦена, Описание)

Покупатели(КодПокупателя, Телефон, Район, Адрес)

Сделки(КодСделки, ДатаСделки, КодТовара, Количество, КодПокупателя)

Вариант 5. Ведение заказов.

Заказчики(КодЗаказчика, Наименование, Адрес, Телефон, КонтактноеЛицо)

Заказы(КодЗаказа, ДатаЗаказа, Скидка, Доставка)

Заказано(КодЗаказа, КодЗаказчика, Количество)

Вариант 6. Бюро по трудоустройству.

Работодатели(КодРаботодателя, ВидДеятельности, Адрес, Телефон)

Сделки (КодСоискателя, КодРаботодателя, Должность, ХарактерРаботы, Комиссионные)

Соискатели (КодСоискателя, Фамилия, Имя, Отчество, Квалификация, ИныеДанные)

Вариант 7. Нотариальная контора.

Клиенты(КодКлиента, Название, ВидДеятельности, Адрес, Телефон)

Сделки(КодСделки, КодКлиента, КодУслуги, Сумма, Комиссионные, Описание)

Услуги(КодУслуги, Название, Описание)

Вариант 8. Фирма по продаже запчастей.

Поставщики(КодПоставщика, Адрес, Телефон)

Детали(КодДетали, Название, Артикул, Цена, Примечание)

Поставки(КодПоставщика, КодДетали, Количество, Дата)
Вариант 9. Курсы по повышению квалификации.
Группы(НомерГруппы, Специальность, Отделение, КоличествоСтудентов)
Преподаватели (КодПреподавателя, Фамилия, Имя, Отчество, Телефон)
Нагрузка(КодПреподавателя, НомерГруппы, КоличествоЧасов, ТипЗаняния, Оплата)
Вариант 10. Определение факультативов для студентов.
Студенты (КодСтудента, Фамилия, Имя, Отчество, Адрес, Телефон)
Предметы(КодПредмета, Название, Объем, Тип)
УчебныйПлан(КодСтудента, КодПредмета, Семестр, Оценка, Преподаватель)

Основная литература

[1-5] – согласно таблице раздела 7.

Дополнительная литература

[6-13] – согласно таблице раздела 7.

9.2. Методические указания по выполнению курсовой работы

Порядок выполнения курсовой работы.

Курсовая работа представляет собой самостоятельно выполненное обучающимся логически завершенное исследование. При выполнении курсовой работы обучающийся должен:

- совершенствовать теоретические знания по дисциплине «Базы данных»;
- продемонстрировать способность обобщать, систематизировать и анализировать информацию, необходимую для проведения исследования и решения поставленных задач;
- приобрести практические навыки в освоении методологии нормализации отношений при создании схемы базы данных для конкретной предметной области;
- приобрести практические навыки проектирования БД;
- приобрести практические навыки по созданию баз данных, схем данных, заполнению и редактированию таблиц данных в выбранной СУБД;
- приобрести практические навыки реализации запросов на SQL.

Для выполнения курсовой работы преподаватель предоставляет обучающемуся в соответствии с номером варианта документ, представляющий собой предметную область для построения базы данных.

Результаты выполнения курсовой работы оформляются в виде пояснительной записки.

Структурные элементы пояснительной записки:

- титульный лист;
- содержание;
- введение;
- основная часть,
- заключение;
- список использованных источников;
- приложения.

Содержание включает последовательно перечисленные наименования всех разделов, подразделов и приложений с указанием номеров страниц.

Во введении формулируются актуальность, цель и задачи курсовой работы, а также указываются методы, применяемые для проектирования базы данных (БД), и название используемой системы управления базами данных (СУБД).

Основная часть курсовой работы должна содержать описание основных этапов создания схемы базы данных для конкретной предметной области. При этом в первом раздел основной части курсовой работы необходимо представить результаты ER-моделирования и нормализации базы данных для предметной области, а во втором разделе основной части должен быть представлен протокол выполнения операции создания базы данных в выбранной СУБД и реализации различных запросов.

В приложении следует представить распечатки диаграмм ER-моделирования, таблиц данных и схемы данных, программного текста и руководство пользователя.

Список использованных источников должен в обязательном порядке содержать учебную, научную и методическую литературу. При выполнении курсовой работы требуется использовать не менее пяти литературных источников.

Реализация базы данных выполняется с помощью выбранной СУБД (или языка программирования, включающего функции работы с БД). Минимальная реализация системы подразумевает создание базы данных и запросов на SQL.

Важнейшим требованием, предъявляемым к курсовой работе, является самостоятельный характер ее выполнения. Оформление пояснительной записки курсовой работы должно осуществляться в строгом соответствии со стандартом ФГБОУ ВО «БрГУ» «Оформление пояснительной записки учебной работы» СМК СТП 1.4-01-2005. Требования к оформлению курсовой

работы представлены в п.3. Пояснительная записка должна быть выполнена аккуратно, без исправлений. Объем Пояснительной записки должен составлять 20–25 страниц.

Пояснительная записка курсовой работы должна быть выполнена аккуратно, без исправлений. Законченная работа представляется преподавателю для проверки. В случае выявления при проверке ошибок и неточностей, обучающийся допускается к защите курсовой работы только после их устранения.

Защита курсовой работы предусматривает вопросы по основным разделам работы с целью выявления уровня сформированных компетенций и самостоятельности выполнения.

10. ПЕРЕЧЕНЬ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, ИСПОЛЬЗУЕМЫХ ПРИ ОСУЩЕСТВЛЕНИИ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО ДИСЦИПЛИНЕ

- Microsoft Windows Professional Russian
- Microsoft Office Russian
- Антивирусное программное обеспечение Kaspersky Security
- Справочно-правовая система «Консультант Плюс»
- Microsoft Access

11. ОПИСАНИЕ МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЙ БАЗЫ, НЕОБХОДИМОЙ ДЛЯ ОСУЩЕСТВЛЕНИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО ДИСЦИПЛИНЕ

Вид занятия	***Наименование аудитории***	***Перечень основного оборудования***	***№ Лк, ПЗ***
1	**3**	**4**	**5**
Лк	Лекционная аудитория (мультимедийный класс)	Интерактивная доска SMART Board 680i2/Unifl, Интерактивный планшет Wacom PL-720, Колонки Microlab Solo-7C, Ноутбук Samsung R610<NP-R610-FS08>, Телевизор плазменный Samsung 63 PS-63A756T1M	Лк № 1-9
ЛР	Дисплейный класс	Системный блок AMD A10-7800 Radeon R7 (12 шт.), Системный блок для слабовидящих пользователей AMD A10-7850K (1 шт.), Монитор Philips233 V5QHABP (13 шт.)	ЛР №№ 1-11
КР	Дисплейный класс	Системный блок AMD A10-7800 Radeon R7 (12 шт.), Системный блок для слабовидящих пользователей AMD A10-7850K (1 шт.), Монитор Philips233 V5QHABP (13 шт.)	
КР	Читальный зал №1	Оборудование 10 ПК i5-2500/H67/4Gb(монитор TFT19 Samsung);	
СР	Читальный зал №1	Оборудование 10 ПК i5-2500/H67/4Gb(монитор TFT19 Samsung); принтер HP LaserJet P2055D	-

Приложение 1

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

1. Описание фонда оценочных средств (паспорт)

№ компетенции	Элемент компетенции	Раздел	Тема	ФОС
ОПК-1	способность использовать нормативно-правовые документы, международные и отечественные стандарты в области информационных систем и технологий	**5**. Жизненный цикл БД	**5.1** Группировки стандартов: по предмету стандартизации, по утверждающей организации, по методическому источнику. **5.2** Стандарты комплекса ГОСТ 34 на создание и развитие АС **5.3** Международный стандарт ISO/IEC 12207: 1995-08-01 на организацию жизненного цикла продуктов программного обеспечения (ПО) **5.4** Содержание основных процессов ЖЦ ИС (ISO/IEC 12207) **5.5** Стадии создания систем (ISO/IEC 15288 System life cycle processes)	вопросы к экзамену 5.1-5.5
ПК-14	способность осуществлять ведение базы данных и поддержку информационного обеспечения решения прикладных задач	**1.** Основные положения теории БД.	**1.1** Информация и данные **1.2** Информационные системы **1.3** Концепция баз данных **1.4** Основные подходы к обработке информации в автоматизированных информационных системах **1.5** Системы управления базами данных **1.6** Классификация баз данных	вопросы к зачету 1.1-1.5
		2. Модели данных.	**2.1** Модели данных. Иерархическая модель данных. Сетевая модель данных. Объектно-ориентированная модель данных. Постреляционная модель данных. **2.2** Модели вычислений **2.3** Реляционные системы и12 правил Кодда **2.4** Нормализация **2.5** Серверы баз данных	вопросы к зачету 2.1-2.5
		3. Реляционные модели и языки.	**3.1** Реляционная алгебра **3.2** Основные и дополнительные операции реляционной алгебры **3.3** Реляционная модель данных **3.4** Целостность БД **3.5** Преимущества реляционной БД **3.6** Преимущества и недостатки СУБД	вопросы к экзамену 3.1-3.6

		5. Жизненный цикл БД.	**5.6** Жизненные циклы информационных систем. **5.7** Жизненный цикл приложения баз данных **5.8** Цели и задачи проектирования **5.9** Подходы к проектированию базы данных **5.10** Этапы проектирования базы данных **5.11** Пользовательские представления	вопросы к экзамену 5.6-5.11
		7. Структурированный язык запросов SQL.	**7.1** Состав и назначение языка SQL **7.2** Реляционные операции. Команды языка манипулирования данными **7.3** Команда SELECT	вопросы к экзамену 7.1-7.3
ПК-6	способность собирать детальную информацию для формализации требований пользователей заказчика	**4.** Предметная область базы данных и ее модели.	**4.1** Понятие предметной области **4.2** Классификация объектов предметной области **4.3** Информационная модель предметной области базы данных **4.4** Функциональная модель предметной области базы данных **4.5** Бизнес-модель процессов (иерархия функций системы) **4.6** Модель потока данных **4.7** Модель жизненного цикла сущности **4.8** Контроль качества результатов анализа предметной области	вопросы к экзамену 4.1-4.8
ПК-7	способность проводить описание прикладных процессов и информационного обеспечения решения прикладных задач	**6.** Проектирование БД.	**6.1** Что такое проектирование базы данных **6.2** Методология концептуального проектирования реляционных баз данных **6.3** Методология логического проектирования реляционных баз данных **6.4** Нормализация и нормальные формы **6.5** Аномалии **6.6** Методология физического проектирования реляционных баз данных	вопросы к экзамену 6.1-6.6

2. Экзаменационные вопросы (вопросы к зачету)

№ п/п	**Компетенции**		**ВОПРОСЫ К ЗАЧЕТУ**	**№ и наименование раздела**
	Код	**Определение**		
1	**2**	**3**	**4**	**5**
1.	ПК-14	способность осуществлять	**1.1** Информация и данные **1.2** Информационные системы	**1.** Основные положения

		ведение базы данных и поддержку информационного обеспечения решения прикладных задач	**1.3** Концепция баз данных	теории БД.
			1.4 Основные подходы к обработке информации в автоматизированных информационных системах	
			1.5 Системы управления базами данных	
			1.6 Классификация баз данных	
			2.1. Модели данных. Иерархическая модель данных. Сетевая модель данных. Объектно-ориентированная модель данных. Постреляционная модель данных.	**2.** Модели данных.
			2.2. Модели вычислений	
			2.3. Реляционные системы и12 правил Кодда	
			2.4. Нормализация	
			2.5. Серверы баз данных	
			3.1. Реляционная алгебра	**3.** Реляционные модели и языки.
			3.2. Основные и дополнительные операции реляционной алгебры	
			3.3. Реляционная модель данных	
			3.4. Целостность БД	
			3.5. Преимущества реляционной БД	
			3.6. Преимущества и недостатки СУБД	

№ п/п	**Компетенции**		**ЭКЗАМЕНАЦИОННЫЕ ВОПРОСЫ**	**№ и наименование раздела**
	Код	**Определение**		
1	**2**	**3**	**4**	**5**
1.	**ОПК-1**	способность использовать нормативно-правовые документы, международные и отечественные стандарты в области информационных систем и технологий	**5.1** Группировки стандартов: по предмету стандартизации, по утверждающей организации, по методическому источнику. **5.2** Стандарты комплекса ГОСТ 34 на создание и развитие АС **5.3** Международный стандарт ISO/IEC 12207: 1995-08-01 на организацию жизненного цикла продуктов программного обеспечения (ПО) **5.4** Содержание основных процессов ЖЦ ИС (ISO/IEC 12207) **5.5** Стадии создания систем (ISO/IEC 15288 System life cycle processes)	**5.** Жизненный цикл БД.
2.	ПК-6	способность собирать детальную информацию для формализации требований пользователей заказчика	**4.1** Понятие предметной области **4.2** Классификация объектов предметной области **4.3** Информационная модель предметной области базы данных **4.4** Функциональная модель предметной области базы данных **4.5** Бизнес-модель процессов (иерархия функций системы) **4.6** Модель потока данных **4.7** Модель жизненного цикла сущности **4.8** Контроль качества результатов анализа предметной области	**4.** Предметная область базы данных и ее модели.

3.	ПК-14	способность осуществлять ведение базы данных и поддержку информационного обеспечения решения прикладных задач	**5.6** Жизненные циклы информационных систем.	**5.** Жизненный цикл БД.
			5.7 Жизненный цикл приложения баз данных	
			5.8 Цели и задачи проектирования	
			5.9 Подходы к проектированию базы данных	
			5.10 Этапы проектирования базы данных	
			5.11 Пользовательские представления	
			7.1 Состав и назначение языка SQL	**7.** Структурированный язык запросов SQL.
			7.2 Реляционные операции. Команды языка манипулирования данными	
			7.3 Команда SELECT	
4.	ПК-7	способностью проводить описание прикладных процессов и информационного обеспечения решения прикладных задач	**6.1** Что такое проектирование базы данных	**6.** Проектирование БД.
			6.2 Методология концептуального проектирования реляционных баз данных	
			6.3 Методология логического проектирования реляционных баз данных	
			6.4 Нормализация и нормальные формы	
			6.5 Аномалии	
			6.6 Методология физического проектирования реляционных баз данных	

3. Описание показателей и критериев оценивания компетенций

Показатели	Оценка	Критерии
Знать *(ОПК-1):* – нормативно-правовые документы, международные и отечественные стандарты в области информационных систем и технологий; *(ПК-14):* – классификацию и модели данных; – основные положений концепции баз данных и принципов построения баз данных; – системы управления БД; – основные предложения языка запросов SQL; *(ПК-6):* – основные направления сбора и анализа требований пользователей; *(ПК-7):* – основные конструкции	**зачтено**	Оценка «**зачтено**» выставляется в случае, если студент демонстрирует: – всестороннее систематическое знание основных положений теории баз данных; – правильное выполнение лабораторных заданий, направленных на использование глубоких теоретических знаний и овладение навыками применения современных методов сбора, обработки и анализа данных; – владеет современными методами проектирования базы данных.
	не зачтено	Оценка **«не зачтено»** выставляется в случае, если студент демонстрирует: – существенные пробелы в знании основных положений теории баз данных; – принципиальные ошибки при выполнении лабораторных заданий, направленных на использование теоретических знаний и овладение навыками применения современных методов сбора, обработки и анализа данных

информационной модели предметной области базы данных; – основные конструкции функциональной модели предметной области, предназначенных для описания процессов обработки информации, **Уметь** *(ОПК-1):* – использовать нормативно-правовые документы, международные и отечественные стандарты в области информационных систем и технологий; *(ПК-14):* – выбирать инструментальные средства и технологии проектирования БД; – применять методы проектирования баз данных, – работать в конкретной СУБД; – работать с базой данных средствами языка SQL *(ПК-6):* – разрабатывать концептуальную модель прикладной области; *(ПК-7):* – использовать методологию моделирования «сущность-связь»; – использовать методологию IDEF0; **Владеть** *(ОПК-1):* – навыками работы с нормативно-правовыми документами, международными и отечественными стандартами в области информационных систем и технологий; *(ПК-14):* – навыками применения современных методов сбора, обработки и анализа данных; – навыками реализации проектирования реляционной БД, – навыками работы в конкретной *(ПК-6):* – навыками построения концептуальной модели предметной области; *(ПК-7):* – навыками ER-моделирования; – навыками работы с инструментальными средствами проектирования баз данных.	**отлично**	Оценка «**отлично**» выставляется в случае, если студент демонстрирует: – всестороннее систематическое знание основных положений теории баз данных; – всестороннее знание стандартов при проектировании базы данных; – правильное выполнение лабораторных заданий, направленных на использование глубоких теоретических знаний и овладение навыками применения современных методов сбора, обработки и анализа данных; – владеет современными методами проектирования базы данных.
	хорошо	Оценка «**хорошо**» выставляется в случае, если студент демонстрирует: – недостаточное полное знание основных положений теории баз данных; – недостаточное знание международных и отечественных стандартов, используемых при проектировании базы данных; – выполнение с несущественными ошибками лабораторных заданий, направленных на использование теоретических знаний и овладение навыками применения современных методов сбора, обработки и анализа данных; – не в полной мере владеет современными методами проектирования базы данных.
	удовлетвори тельно	Оценка «**удовлетворительно**» выставляется в случае, если студент демонстрирует: – частичное знание основных положений теории баз данных; – частичное знание международных и отечественных стандартов, используемых при проектировании базы данных; – частичное выполнение ошибками лабораторных заданий, направленных на использование теоретических знаний и овладение навыками применения современных методов сбора, обработки и анализа данных; – не в полной мере владеет современными методами проектирования базы данных.
	неудовлетво рительно	Оценка «**неудовлетворительно**» выставляется в случае, если студент демонстрирует: – существенные пробелы в знании основных положений теории баз данных; – существенные пробелы в знании основных стандартов в области информационных систем и технологий; – принципиальные ошибки при выполнении лабораторных заданий, направленных на использование теоретических знаний и овладение навыками применения современных методов сбора, обработки и анализа данных. – не владеет современными методами проектирования базы данных.

4. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и опыта деятельности

Цель и задачи дисциплины Б1.Б.Б18 Базы данных представлены в разделе 1 настоящей рабочей программы. Место дисциплины в структуре образовательной программы представлено в разделе 2 настоящей рабочей программы. Распределение объема дисциплины по формам обучения с указанием видов учебных занятий представлено в разделе 3 настоящей рабочей программы. Содержание дисциплины указано в разделе 4 настоящей рабочей программы.

Учебно-методические материалы для самостоятельной работы студентов по дисциплине находятся в свободном доступе в соответствии с разделом 6 настоящей рабочей программы.

При изучении дисциплины необходимо использовать литературу, указанную в разделе 7 настоящей рабочей программы, а также перечень ресурсов информационно-телекоммуникационной сети «Интернет», представленных в разделе 8 настоящей рабочей программы.

Консультации для студентов по дисциплине проводятся в соответствии с графиком проведения консультаций, представленном на стенде кафедры, за которой закреплена указанная дисциплина.

К зачету, экзамену допускаются студенты очной формы обучения, которые выполнили, оформили и защитили все лабораторные работы, предусмотренные в конкретном семестре. Методические указания по выполнению и оформлению представлены в разделе 9.1. настоящей рабочей программы.

К экзамену допускаются студенты заочной формы обучения, которые теоретически подготовлены по всем вопросам и владеют навыками проектирования базы данных.

Информационные технологии, используемые при освоении дисциплины, перечислены в разделе 10 настоящей рабочей программы.

Оценка знаний, умений, навыков осуществляется в процессе промежуточной аттестации обучающихся по дисциплине, которая осуществляется в виде экзамена. Для оценивания знаний, умений, навыков используются ФОС по дисциплине.

Зачет, экзамен проводятся в устной форме по выданному преподавателем заданию.

По итогам выполненного задания преподаватель оценивает уровень знаний, умений, навыков. Описание показателей и критериев оценивания компетенций, сформированных по итогам изучения дисциплины, представлено в разделе 3 Приложения 1 настоящей рабочей программы. Основными оценочными средствами при проведении промежуточной аттестации являются вопросы к экзамену.

Приложение 2

АННОТАЦИЯ
рабочей программы дисциплины
Базы данных

1. Цель и задачи дисциплины

Цель дисциплины

- овладение обучающимися основами теоретических и практических знаний в области баз данных, современных инструментальных средств моделирования и управления доступом к информационным массивам

Основными задачами дисциплины являются:

изучение теоретических и практических основ применения возможностей баз данных (проектирование, ведение и использование баз данных).

2. Структура дисциплины

2.1 Распределение трудоемкости по отдельным видам учебных занятий, включая самостоятельную работу: 35 ч. лекции, 53ч. лабораторные работы, самостоятельная работа 101 ч.

Общая трудоемкость дисциплины составляет 216 часа, 6 зачетных единицы

2.2 Основные разделы дисциплины:

1 - Основные положения теории БД.
2 - Модели данных.
3 - Реляционные модели и языки.
4 - Предметная область базы данных.
5 - Жизненный цикл БД.
6 - Проектирование БД.
7 - Структурированный язык запросов SQL.

3. Планируемые результаты обучения (перечень компетенций)

Процесс изучения дисциплины направлен на формирование следующих компетенций:

ОПК-1 - способность использовать нормативно-правовые документы, международные и отечественные стандарты в области информационных систем и технологий;

ПК-14 - способность осуществлять ведение базы данных и поддержку информационного обеспечения решения прикладных задач;

ПК-6 - способностью собирать детальную информацию для формализации требований пользователей заказчика;

ПК-7 - способностью проводить описание прикладных процессов и информационного обеспечения решения прикладных задач.

4. Вид промежуточной аттестации: зачет, экзамен

Приложение 3

Протокол о дополнениях и изменениях в рабочей программе
на 20___-20___ учебный год

1. В рабочую программу по дисциплине вносятся следующие дополнения:

_____

_____

2. В рабочую программу по дисциплине вносятся следующие изменения:

_____

_____

_____

Протокол заседания кафедры №____ от «___» _____ 20 __ __ г.,
(разработчик)

Заведующий кафедрой _____ _____
(подпись) *(Ф.И.О.)*

Программа составлена в соответствии с федеральным государственным образовательным стандартом высшего образования по направлению подготовки 09.03.03 Прикладная информатика от «12» марта 2015 г. № 207

для набора 2017 года: и учебным планом ФГБОУ ВО «БрГУ» для очной формы обучения от «06» марта 2017 г. №125, заочной формы обучения от «06» марта 2017 г. №125

для набора 2018 года и учебным планом ФГБОУ ВО «БрГУ» для очной формы обучения от «12» марта 2018 г. № 130, заочной формы обучения от «12» марта 2018 г. № 130

Программу составил:

Вахрушева М.Ю., доцент баз. МиИТ, доцент, к.физ.-мат.н. _____

Рабочая программа рассмотрена и утверждена на заседании базовой кафедры МиИТ

от «19» декабря 2018 г., протокол № 8

И.о. заведующего базовой кафедрой МиИТ _____ Е.И. Луковникова

СОГЛАСОВАНО:

И.о. заведующего выпускающей базовой кафедрой МиИТ _____ Е.И. Луковникова

Директор библиотеки _____ Т.Ф. Сотник

Рабочая программа одобрена методической комиссией факультета ФЭиУ

от «28» декабря 2018 г., протокол № 4

Председатель методической комиссии факультета _____ Е.В. Трапезникова

СОГЛАСОВАНО:

Начальник учебно-методического управления _____ Г.П. Нежевец

Регистрационный №_____