ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

УПРАВЛЕНИЕ ДИСТАНЦИОННОГО ОБУЧЕНИЯ И ПОВЫШЕНИЯ КВАЛИФИКАЦИИ

Кафедра «Эксплуатация транспортных систем и логистика»

Учебное пособие «Использование баз данных на транспорте. Технология создания»

по дисциплине

«Базы данных на транспорте»

Авторы

Гальченко Г.А., Марченко Ю.В., Попов С.И.

Ростов-на-Дону, 2018

Аннотация

Данное учебное пособие предназначено для студентов очной и заочной форм обучения направлений 23.02.01 «Технология транспортных процессов» и 05.10.00 «Профессиональное обучение».

Изложен краткий теоретический курс, примеры использования баз данных при решении транспортных задач, основные сведения о системах управления базами данных, тесты, практические и лабораторные работы.

Авторы

К.ф.-м.н., доцент кафедры «Эксплуатация транспортных систем и логистика» Гальченко Г.А.,
к.т.н., доцент кафедры «Эксплуатация транспортных систем и логистика» Марченко Ю.В.
к.т.н., доцент кафедры «Эксплуатация транспортных систем и логистика» Попов С.И.

Оглавление

ВВЕДЕНИЕ

Информатизация общества ставит перед учебными заведениями проблему воспитания у студентов способностей самостоятельно и творчески использовать средства информатики и информационных технологий в решении учебных и в дальнейшем профессиональных задач. Студенты должны иметь мировоззрение в соответствии с системно-информационной картиной мира, знать особенности программных средств для будущей профессиональной деятельности, уметь работать с профессиональным программным обеспечением, знать о возможностях применения программных средств в будущей деятельности.

Владение информационно-коммуникационными технологиями (ИКТ) – важная составляющая квалификации и средство социальной мобильности. В образовании главным стимулом обучения ИКТ является как решение сегодняшних проблем, так и требование будущего развития. *Современное информационное общество* выдвигает новые требования к формированию информационных культурных ресурсов. Эффективность работы, например, транспортных и технологических машин, оборудования и их подсистем – сервиса и технической эксплуатации – зависит от компетентности специалистов в области использования средств автоматизации сбора, поиска, использования информации об обеспечении качества ремонта и технологического обслуживания техники организациями технического сервиса с использованием средств ИТ. Ознакомление студентов с инновационными процессами в области ИТ, использование прикладных программных продуктов в целях повышения профессионального уровня – одна из главных задач системы подготовки специалистов, их успешной профессиональной деятельности. Использование в управлении информационных технологий является непременным условием повышения эффективности управленческого труда.

Учебное пособие написано с учетом государственных требований к минимуму содержания и уровню подготовки выпускников высших учебных заведений, а также учитывает обязательный минимум содержания образования по информатике, рекомендованный Министерством образования РФ.

Учебное пособие предназначено для углубленного изучения работы в системах управления базы данных (БД) с использованием соответствующих программных средств

обработки данных. Изучив материал данного учебного пособия, студент будет способен:

– понимать сущность и социальную значимость своей будущей профессии, проявлять к ней устойчивый интерес;

– использовать информационно-коммуникационные технологии для совершенствования своей профессиональной деятельности;

– определять адекватные способы решения учебной задачи на основе заданных алгоритмов; комбинировать известные алгоритмы деятельности в ситуациях, не предполагающих стандартное применение одного из них.

1. ИСТОРИЯ СОЗДАНИЯ БАЗЫ ДАННЫХ

С середины 60-х гг. до 1980 г. была разработана иерархическая модель данных, а затем появилась сетевая модель данных, которая являлась логическим продолжением иерархической модели. В 1970 г. британский ученый Э.Ф. Кодд выпустил работу по реляционному хранению данных. После выхода работы в печать начинаются активные работы по разработке данной системы хранения информации.

Система управления базами данных (СУБД) выпущена в начале 80-х гг.

Основы реляционных баз данных включают в себя три аспекта:

– структурный – данные представляют из себя наборы отношений;

– целостности – наборы отношений отвечают требованиям целостности;

– обработки – поддерживаются операторы манипулирования отношениями.

В реляционных базах данных поддерживаются принципы нормализации.

в настоящее время активно разрабатываются объектно-ориентированные базы данных, объектно-реляционные.

На сегодняшний день известно более двух десятков форматов данных настольных СУБД, однако наиболее популярными, исходя из числа проданных копий, считаются *dBase*, *Paradox*, *FoxPro* и *Access*. Отмечают также СУБД *Microsoft Data Engine* – по существу серверную СУБД, представляющую

собой «облегченную» версию *Microsoft SQL Server*, но предназначенную для использования главным образом в настольных системах и небольших рабочих группах (табл. 1.1).

Таблица 1.1

Производители СУБД

СУБД	Производитель
Visual dBase	DBase, Inc
Paradox	Corel
Microsoft Access 2007	Microsoft
Microsoft FoxPro	Microsoft
Microsoft Visual FoxPro	Microsoft
Microsoft Visual FoxPro	Microsoft
Microsoft Data Engine	Microsoft

1.1. DBase и Visual dBase

Первая промышленная версия СУБД *dBase* – *dBase II* появилась в начале 80-х годов. Благодаря простоте в использовании, нетребовательности к ресурсам компьютера и, что не менее важно, грамотной маркетинговой политике компании-производителя, этот продукт приобрел немалую популярность, а с выходом следующих его версий – *dBase III* и *dBase III Plus* (1986 г.), оснащенных весьма комфортной по тем временам средой разработки и средствами манипуляции данными, быстро занял лидирующие позиции среди настольных СУБД и средств создания использующих их приложений.

Хранение данных в *dBase* основано на принципе «одна таблица – один файл» (эти файлы обычно имеют расширение *.*dbf*). *MEMO*-поля и *BLOB*-поля (доступные в поздних версиях *dBase*) хранятся в отдельных файлах (обычно с расширением *.*dbt*). Индексы для таблиц также хранятся в отдельных файлах.

Формат данных *dBase* является открытым, что позволило ряду других производителей заимствовать его для создания *dBase*-подобных СУБД, частично совместимых с *dBase* по форматам данных.

После покупки *dBase* компанией *Borland* этот продукт, получивший впоследствии название *Visual dBase*, приобрел набор дополнительных возможностей, характерных для средств разработки этой компании, и для имевшейся у нее другой настольной СУБД – *Paradox*. Среди этих возможностей можно

отметить специальные типы полей для графических данных, поддерживаемые индексы, хранение правил ссылочной целостности внутри самой базы данных, а также возможность манипулировать данными других форматов, в частности серверных СУБД, за счет использования *BDE API* и *SQL Links*.

В настоящее время версия *Visual dBase 7.5* имеет следующие возможности:

– средства манипуляции данными *dBase* и *FoxPro* всех версий;

– средства создания форм, отчетов и приложений;

– средства публикации данных в *Internet* и создания *Web*-клиентов;

– ядро доступа к данным *Advantage Database Server* фирмы *Extended Systems* и *ODBC* – драйвер для доступа к данным этой СУБД;

– средства публикации отчетов в *Web*;

– средства визуального построения запросов;

– средства генерации исполняемых файлов и дистрибутивов.

В настоящее время к *Visual dBase* в качестве дополнения может быть приобретен компонент *dConnections*, позволяющий осуществить доступ к данным *Oracle*, *Sybase*, *Informix*, *MS SQL Server*, *DB2*, *InterBase* из *Visual dBase 7.5* и приложений, созданных с его помощью.

1.2. Paradox

Paradox был разработан компанией *Ansa Software*, и первая его версия увидела свет в 1985 году. Принцип хранения данных в *Paradox* сходен с принципами хранения данных в *dBase* – каждая таблица хранится в своем файле (расширение *.*db*), *MEMO*- и *BLOB*-поля хранятся в отдельном файле (расширение *.*md*) как и индексы (расширение *.*px*). Однако, в отличие от *dBase*, формат данных *Paradox* не является открытым, поэтому для доступа к данным этого формата требуются специальные библиотеки.

Отсутствие «открытости» формата данных имеет свои достоинства. Так как в этой ситуации доступ к данным осуществляется только с помощью «знающих» этот формат библиотек, простое редактирование подобных данных, по сравнению с данными открытых форматов типа *dBase,* существенно затруднено. В этом случае возможны такие недоступные при использовании «открытых» форматов данных

сервисы, как защита таблиц и отдельных полей паролем, хранение некоторых правил ссылочной целостности в самих таблицах – все эти сервисы предоставляются *Paradox*, начиная с первых версий этой СУБД.

По сравнению с аналогичными версиями *dBase* ранние версии *Paradox* обычно предоставляли разработчикам баз данных существенно более расширенные возможности, такие как использование деловой графики в *DOS*-приложениях, обновление данных в приложениях при многопользовательской работе, визуальные средства построения запросов, на основе интерфейса *QBE – Query by Example*, средства статистического анализа данных, а также средства визуального построения интерфейсов пользовательских приложений с автоматической генерацией кода на языке программирования *PAL* (*Paradox Application Language*).

Windows-версии СУБД *Paradox*, помимо перечисленных выше сервисов, позволяют также манипулировать данными других форматов, в частности *dBase* и данными, хранящимися в серверных СУБД.

Версия данной СУБД – *Paradox 9* – поставляется в двух вариантах: *Paradox 9 Standalone Edition* и *Paradox 9 Developer's Edition*. Первый из них предназначен для использования в качестве настольной СУБД и входит в *Corel Office Professional*, второй – в качестве как настольной СУБД, так и средства разработки приложений и манипуляции данными в серверных СУБД. Обе версии содержат:

– средства манипуляции данными *Paradox* и *dBase*;

– средства создания форм, отчетов и приложений;

– средства визуального построения запросов;

– средства публикации данных и отчетов в *Internet* и создания *Web*-клиентов;

– *Corel Web* – сервер;

– *ODBC* – драйвер для доступа к данным формата *Paradox* из *Windows*-приложений;

– средства для доступа к данным формата *Paradox* из *Java*-приложений.

Помимо этого, *Paradox 9 Developer's Edition* содержит:

– *Run – time* – версия *Paradox* для поставки вместе с приложениями;

– средства создания дистрибутивов;

– драйверы *SQL Links* для доступа к данным серверных СУБД.

В последнее время популярность этого продукта как

средства разработки несколько снизилась, хотя в мире эксплуатируется еще немало информационных систем, созданных с его помощью.

1.3. Microsoft Access

Первая версия СУБД *Access* появилась в начале 90-х годов. Это была первая настольная реляционная СУБД для 16-разрядной версии *Windows*. Популярность *Access* значительно возросла после включения этой СУБД в состав *Microsoft Office* (рис. 1.1).

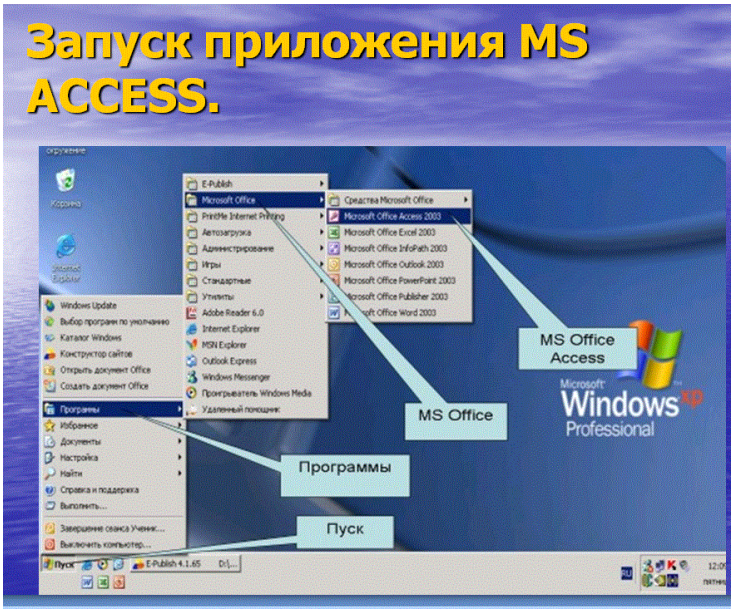


Рис. 1.1 Запуск приложений MS Access

В отличие от *Visual FoxPro Access* ориентирован в первую очередь на пользователей *Microsoft Office*, в том числе и не знакомых с программированием. Версия этой СУБД – *Access 2007* входит в состав *Microsoft Office 2007 Professional* и *Premium*, а также доступна как самостоятельный продукт. В состав *Access 2007* входят:

– средства манипуляции данными *Access* и данными, доступными через *ODBC* (последние могут быть «присоединены» к базе данных *Access*);

– средства создания форм, отчетов и приложений; при этом

отчеты могут быть экспортированы в формат *Microsoft Word* или *Microsoft Excel*, а для создания приложений используется *Visual Basic for Applications*, общий для всех составных частей *Microsoft Office*;

– средства публикации отчетов в *Internet*;

– средства создания интерактивных *Web*-приложений для работы с данными (*Data Access Pages*);

– средства доступа к данным серверных СУБД через *OLE DB*;

– средства создания клиентских приложений для *Microsoft SQL Server*;

– средства администрирования *Microsoft SQL Server*.

Поддержка *COM* в *Access* выражается в возможности использовать элементы управления *ActiveX* в формах и *Web*-страницах, созданных с помощью *Access*. В отличие от *Visual FoxPro* создание *COM*-серверов с помощью *Access* не предполагается. Иными словами, *Microsoft Access* может быть использован, с одной стороны, в качестве настольной СУБД и составной части офисного пакета, а с другой, – в качестве клиента *Microsoft SQL Server*, позволяющего осуществлять его администрирование, манипуляцию его данными и создание приложений для этого сервера.

Помимо манипуляции данными *Microsoft SQL Server*, *Access 2007* позволяет также в качестве хранилища данных использовать *Microsoft Data Engine* (*MSDE*), представляющий собой по существу настольный сервер баз данных, совместимый с *Microsoft SQL Server*.

2. ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ БАЗЫ ДАННЫХ НА ТРАНСПОРТЕ

2.1. Цель создания базы данных

Цель создания базы данных – помочь людям и организациям вести учет и осуществлять быстрый поиск нужной информации в зашифрованном виде. При использовании программного обеспечения (ПО), содержащего базу данных, работа займет несколько секунд. При этом полностью исключив ошибки. Программное обеспечение, включающее в себя базу данных предприятия, – это бережное

хранение и систематизация всей важной информации, быстрый поиск, автоматический расчет, представление аналитического материала для улучшения бизнес-процессов, исключение ошибок при обработке введенных данных.

Рассмотрим некоторое автопредприятие, имеющее определенный парк автомобилей, которые выезжают на рейсы (рис. 2.1). Каждый рейс выполняет определенный водитель, о котором известны определенные сведения. О каждом автомобиле известные некоторые данные, модель, цвет, тип двигателя и т.д. Необходимо хранить всю эту информация, осуществлять быстрый поиск необходимых данных, составлять отчеты.

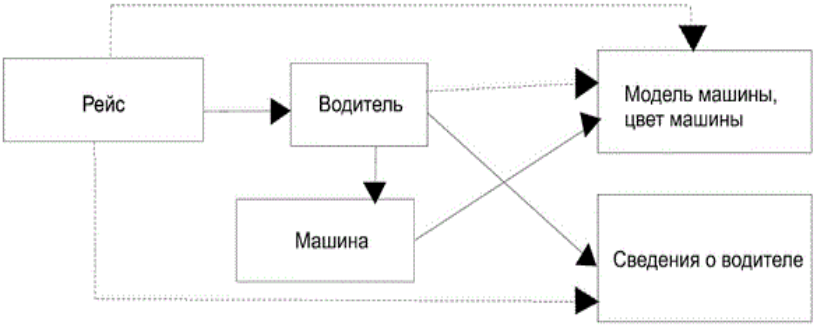


Рис. 2.1. Схема работы автопредприятия

Основные элементы БД это таблицы, запросы, формы и отчеты. Созданная БД обычно содержит:

– структуру спроектированных таблиц;
– схему данных со связями между таблицами;
– примеры форм, обеспечивающих интерфейс пользователя;
– запросы (в режиме Конструктора и на языке SQL);
– отчеты (в режиме отчета и в режиме Конструктора).

2.2. Примеры базы данных по автомобилям, городскому автотранспорту и маршрутным такси

База данных по автомобилям MAZDA и HYNDAI представлена на рис. 2.2, 2.3.

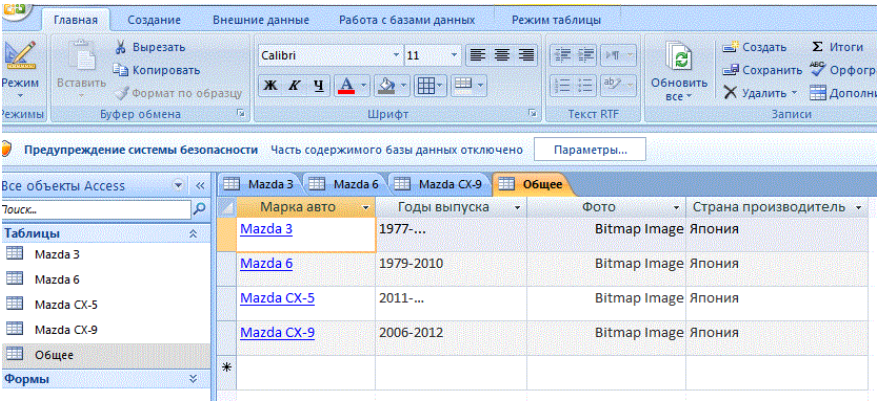


Рис. 2.2. БД автомобиля Mazda

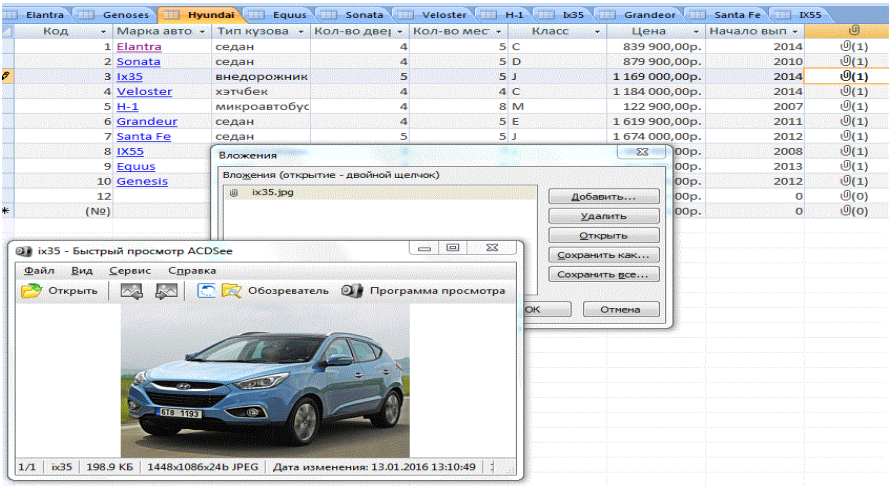


Рис. 2.3. БД автомобиля Hyundai

База данных «Маршрутное такси». Автотранспортное предприятие осуществляет перевозку пассажиров по городским маршрутам на маршрутных такси (рис. 2.4, 2.5). Рейсы на маршрутах осуществляют водители на автобусах (рис. 2.6–2.18). Рейс характеризуется временем начала и окончания движения, количеством проданных билетов. Цена билета устанавливается отдельно для каждого маршрута. Автобусы характеризуются количеством мест, годом выпуска.

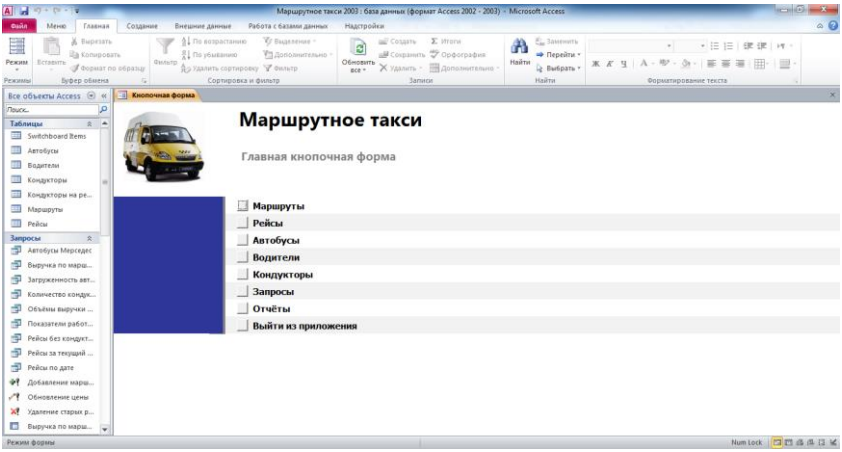


Рис. 2.4. Форма базы данных «Маршрутное такси»

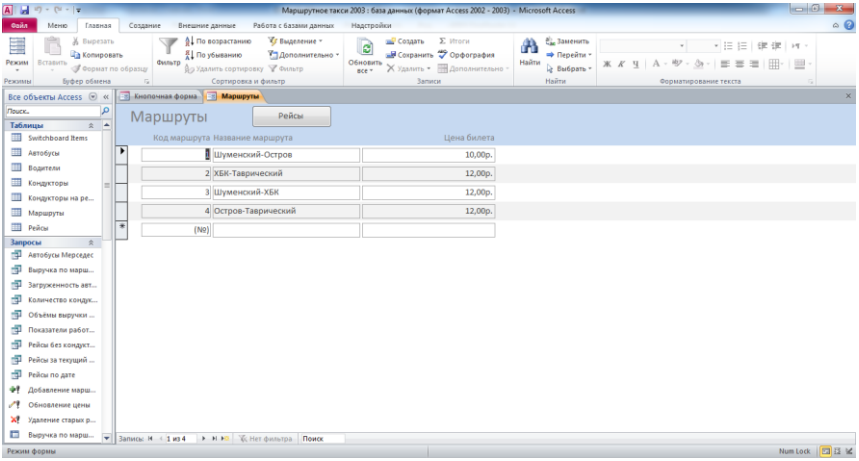


Рис. 2.5. Форма «Маршруты»

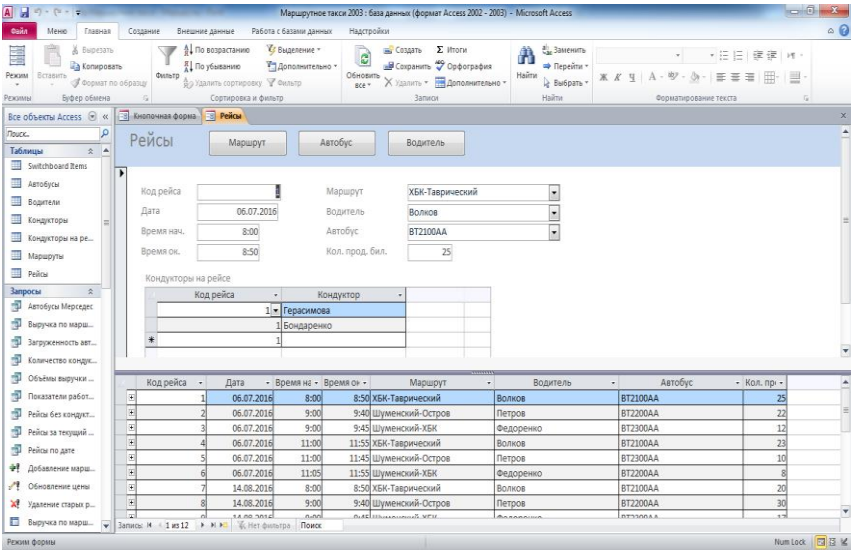


Рис. 2.6. Форма «Рейсы»

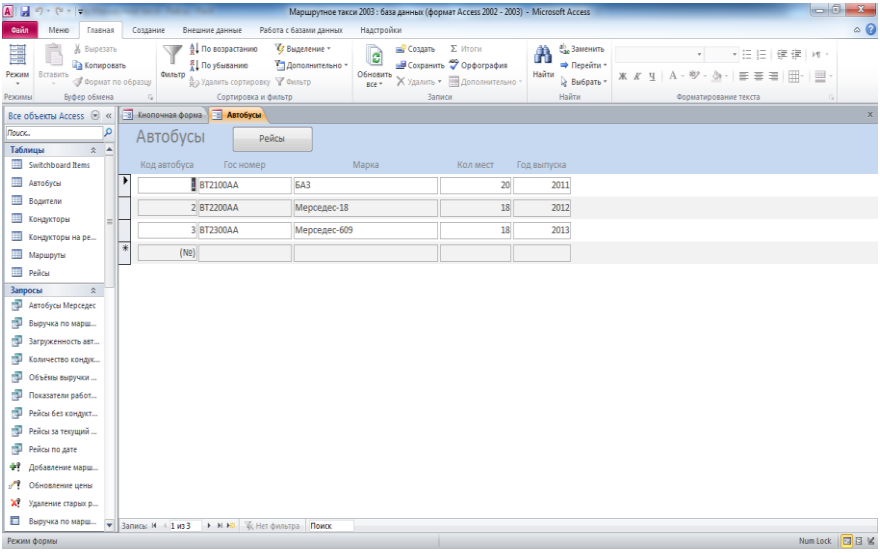


Рис. 2.7. Форма «Автобусы»

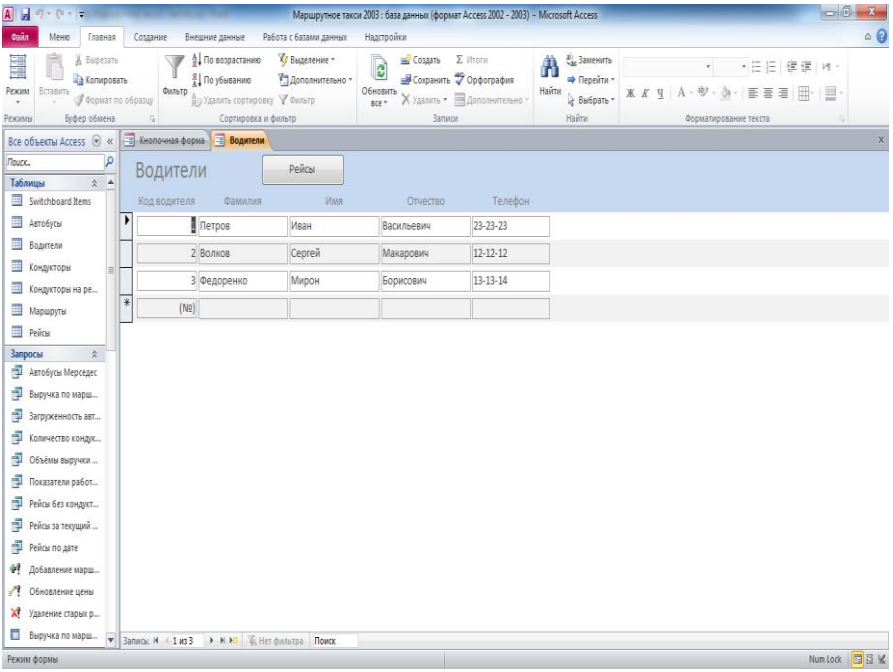


Рис. 2.8. Форма «Водители»

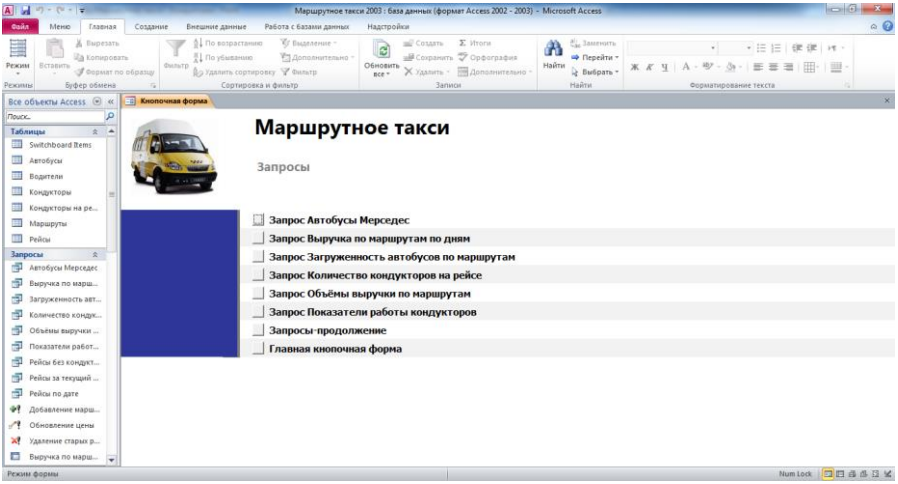


Рис. 2.9. Кнопочная форма БД «Маршрутное такси» – «Запросы»

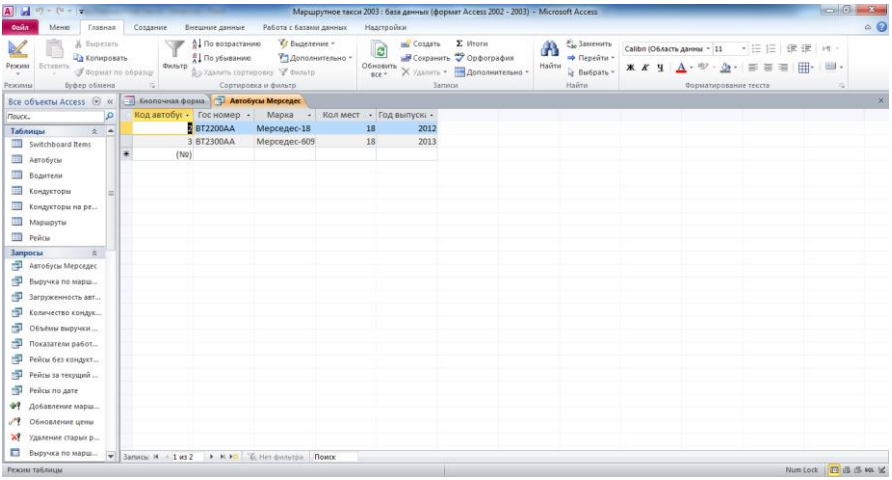


Рис. 2.10. Запрос «Автобусы Мерседес»

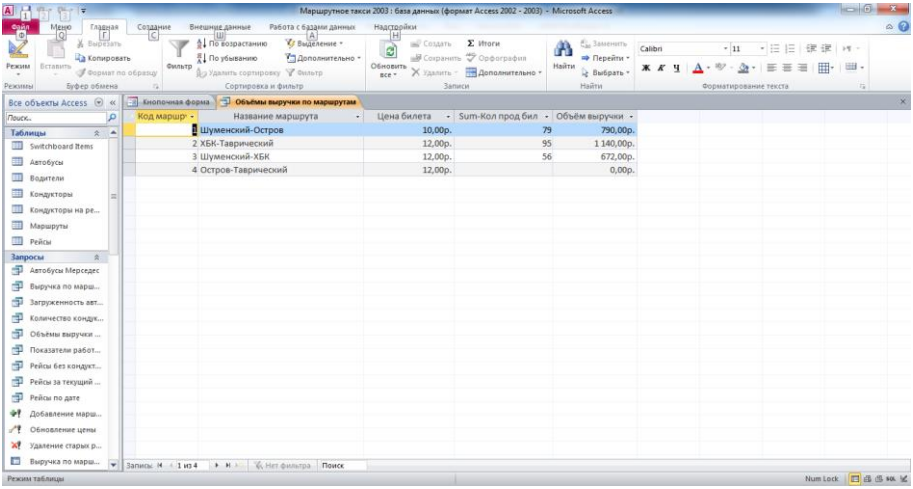


Рис. 2.11. Запрос «Объёмы выручки по маршрутам»

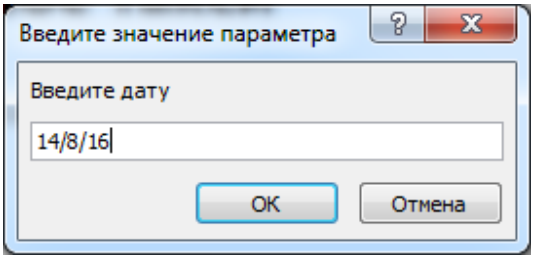


Рис. 2.12. Окно для ввода параметра «Дата»

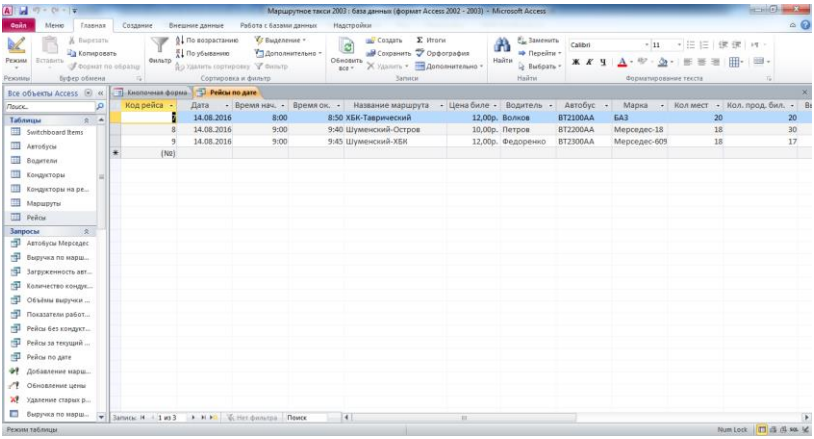


Рис. 2.13. Запрос «Рейсы по дате»

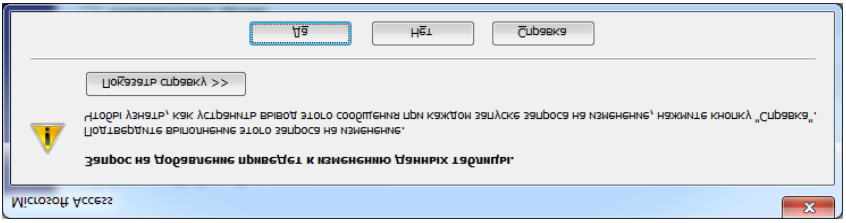


Рис. 2.14. Сообщение о выполнении запроса на добавление. Для подтверждения нажать «Да» и следовать инструкциям

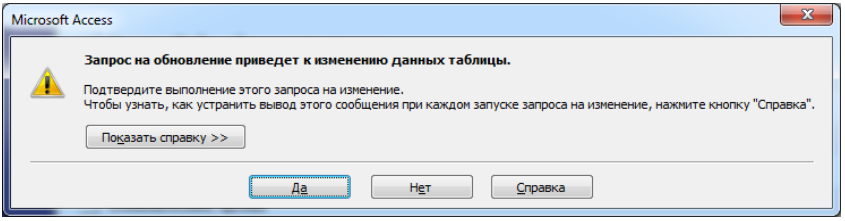


Рис. 2.15. Сообщение о выполнении запроса на обновление. Для подтверждения нажать «Да» и следовать инструкциям

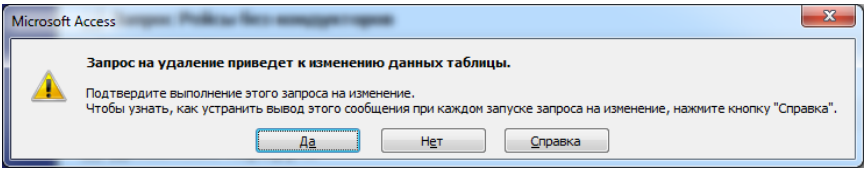


Рис. 2.16. Сообщение о выполнении запроса на удаление. Для подтверждения нажать «Да» и следовать инструкциям

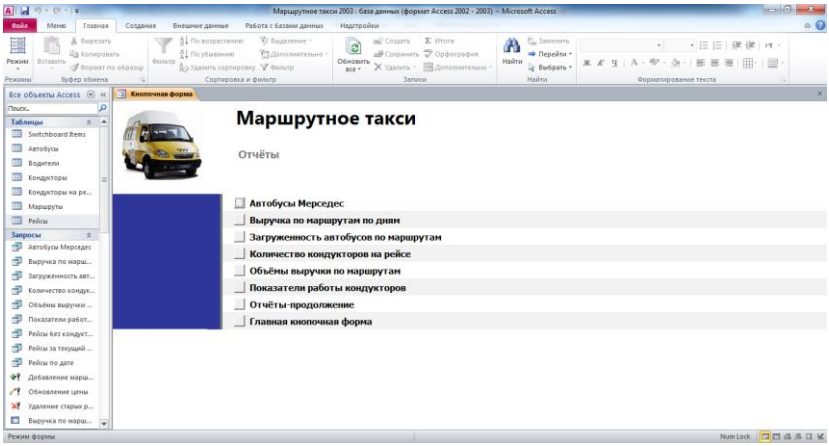


Рис. 2.17. Форма базы данных «Маршрутное такси» – «Отчёты»

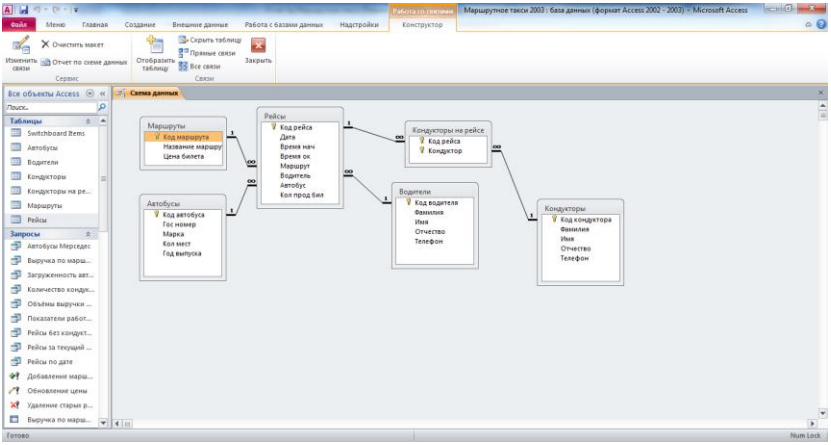


Рис. 2.18. Схема данных готовой базы данных «Маршрутное такси» отображает связи таблиц «Маршруты», «Автобусы», «Рейсы», «Водители»

Каждая таблица необходима для выполнения конкретных функций. В таблицу «Заказы» заносятся все сведения о поступивших заказах: когда и от кого прибыл, какой диспетчер принял, и какой водитель будет исполнять заказ, имя и контактный телефон абонента, места отправления и назначения, статус и стоимость. На основании данных этой таблицы формируются все отчеты данного приложения: Сводный отчет по диспетчерам, Сводный отчет по водителю, «Чёрный список» и отчет диспетчера за смену. Атрибуты и их домены показаны на

табл. 2.1.

Таблица 2.1 «Заказы»

Наименование атрибута	Тип данных	Описание
Номер заказа	Счетчик	Регистрирует порядковый номер заказа
Дата и время поступления заказа	Дата/Время	Генерируется автоматически
Дата и время назначения заказа	Дата/Время	Время, на которое назначен заказ
Место отправления	Текстовый	Вводится вручную
Место назначения	Текстовый	Вводится вручную
Абонент	Текстовый	Вводится вручную
Телефон абонента	Тестовый	Так как при заполнении используются знаки препинания
Стоимость заказа	Денежный	
Фамилия водителя	Текстовый	Необходима для формирования сводных отчетов
Диспетчер	Текстовый	Не отображается, но необходима для формирования сводных отчетов
Статус	Текстовый	Принимает одно из трех значений: «Выполняется», «Выполнен», «Отменен»

Таблицы «Водители» и «Диспетчеры» выполняют схожие функции – это хранение данных о сотрудниках, работающих в данном таксопарке, естественно данные различаются по специфике выполняемых операций (табл. 2.2, 2.3). Например, у водителей существуют поля, в которых находятся сведения о транспортном средстве, а у диспетчеров наличие полей «Login» и «Password», так как они необходимы для входа в систему.

Таблица 2.2 «Водители»

Наименование атрибута	Тип данных	Описание
Позывной	Числовой	Идентификационный номер
Фамилия	Текстовый	Фамилия водителя
Имя	Текстовый	Имя водителя
Отчество	Текстовый	Отчество водителя
Дата рождения	Дата/время	Дата рождения водителя
Серия паспорта	Числовой	Серия паспорта водителя
Номер паспорта	Числовой	Номер паспорта водителя
Кем выдан паспорт	Текстовый	УВД выдавшее паспорт
Дата выдачи	Дата/время	Когда был выдан паспорт
Адрес	Текстовый	Где фактически проживает водитель
Марка автомобиля	Текстовый	Марка автомобиля водителя
Номер автомобиля	Текстовый	Номер регистрации в гаи
Цвет	Текстовый	Цвет автомобиля водителя

Таблица 2.3 «Диспетчеры»

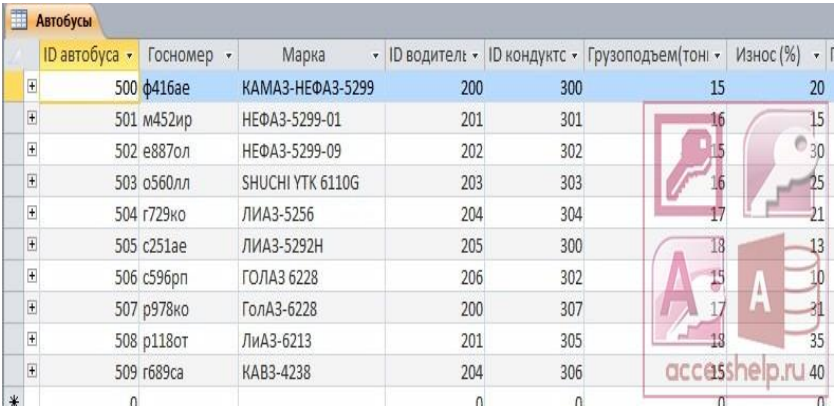
Наименование атрибута	Тип данных	Описание
Табельный номер	Числовой	Идентификационный номер
Фамилия	Текстовый	Фамилия диспетчера
Имя	Текстовый	Имя диспетчера
Отчество	Текстовый	Отчество диспетчера
Дата рождения	Дата/время	Дата рождения диспетчера
Серия паспорта	Числовой	Серия паспорта диспетчера
Номер паспорта	Числовой	Номер паспорта диспетчера
Кем выдан паспорт	Текстовый	УВД выдавшее паспорт
Дата выдачи	Дата/время	Когда был выдан паспорт
Адрес	Текстовый	Где фактически проживает диспетчер
Login	Текстовый	Ник для входа в программу
Password	Текстовый	Индивидуальный код для входа в программу

Таблица «Водители_Смены» необходима для регистрации водителей прибывших на смену, при начале работы диспетчер заносит в эту таблицу водителей прибывших на конкретную смену, из списка всех существующих водителей. Она состоит из четырех атрибутов, представленных на табл. 2.4.

Таблица 2.4 «Водители_Смены»

Наименование атрибута	Тип данных	Описание
ID	Счетчик	Номер смены
Фамилия	Текстовый	Фамилия водителя прибывшего на смену
Имя	Текстовый	Имя водителя прибывшего на смену
Позывной	Числовой	Идентификационный номер прибывшего на смену водителя

База данных «Городской транспорт» предназначена для автоматизации работы компании, занимающейся городскими перевозками. В базе таблицы заполнены данными, выполнены простые и перекрестные запросы, а также запросы на добавление, обновление и удаление. Также сделаны формы для работы с данными и отчеты, которые можно выводить на печать. База данных *Access* Автосалон содержит 5 таблиц, 11 запросов, 6 форм + главная кнопочная форма, 4 отчета.



Автобусы

ID автобуса	Госномер	Марка	ID водител	ID кондуктс	Грузоподъем(тонн	Износ (%)
500	ф416ае	КАМАЗ-НЕФАЗ-5299	200	300	15	20
501	м452ир	НЕФАЗ-5299-01	201	301	16	15
502	е887ол	НЕФАЗ-5299-09	202	302	15	30
503	о560лл	SHUCHI YTK 6110G	203	303	16	25
504	г729ко	ЛИАЗ-5256	204	304	17	21
505	с251ае	ЛИАЗ-5292Н	205	300	18	13
506	с596рп	ГОЛАЗ 6228	206	302	15	10
507	р978ко	ГолАЗ-6228	200	307	17	31
508	р118от	ЛиАЗ-6213	201	305	18	35
509	г689са	КАВЗ-4238	204	306	15	40
0			0	0	0	0

Рис. 2.19. Таблица БД «Автобусы»

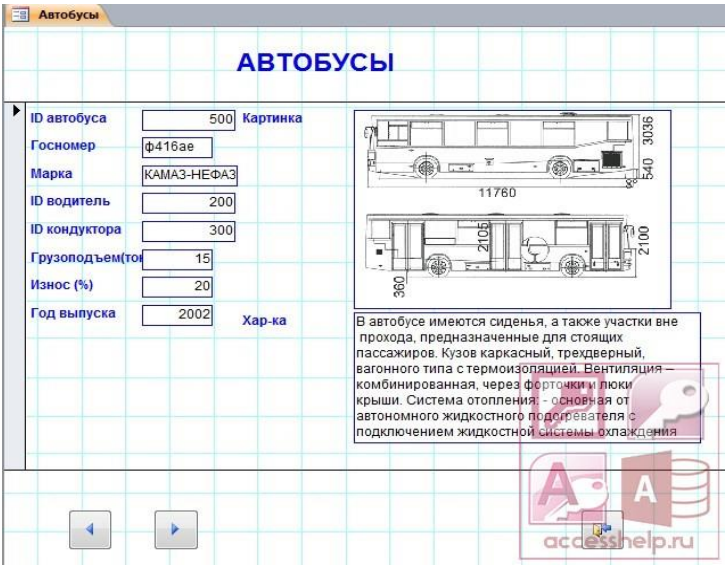


Рис. 2.20. Форма «Автобусы»

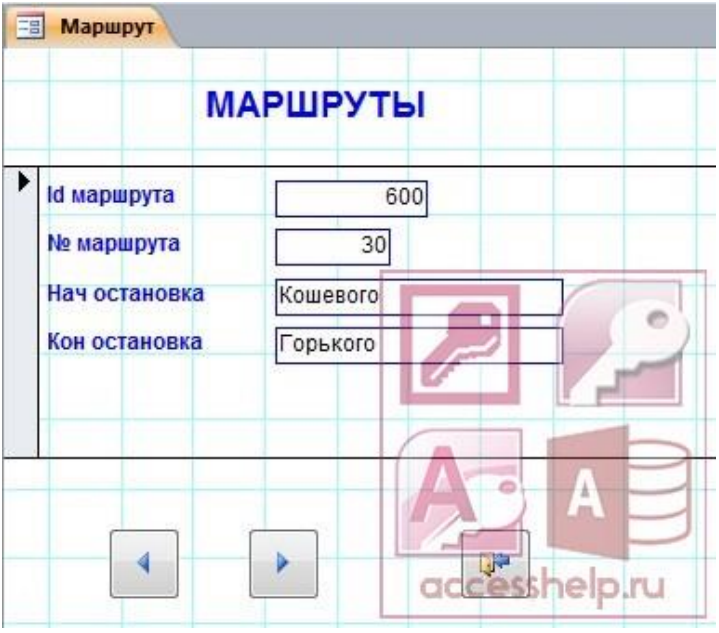


Рис. 2.21. Форма «Маршруты»

Рис. 2.22. Форма «Пассажирские перевозки»

База данных «Городской транспорт» позволяет добавлять и редактировать информацию о маршрутах, автобусах, водителях и кондукторах (рис. 2.19, 2.20). Также в базе данных «Городской транспорт» предусмотрены запросы на вывод определенного номера маршрута, определенного водителя и кондуктора, стажа и общего оклада каждого сотрудника и т.д. (рис. 2.21, 2.22). Реализован запрос на обновление, запрос на удаление, запрос на добавление, на создание таблицы, перекрестный.

3. ТЕХНОЛОГИЯ ХРАНЕНИЯ, ПОИСКА И СОРТИРОВКИ ИНФОРМАЦИИ В БАЗАХ ДАННЫХ

В данной главе авторы предлагают познакомиться с основными принципами организации и обработки больших массивов данных об объектах и явлениях реального мира, изучить виды информационных систем, моделей данных, принципы организации подхода хранения данных на примере системы управления базами данных *MS Access*.

3.1. Информационные системы

Информационные системы (ИС) – большие массивы данных вместе с программно-аппаратными средствами для их обработки называются. Основа любой информационной системы – база данных (БД). База данных – это некоторое подобие электронной картотеки, электронного хранилища данных, которое хранится в компьютере в виде одного или нескольких файлов (рис. 3.1).

Рис. 3.1. Электронная регистратура

Информационные системы можно классифицировать по масштабу и способу организации.

По масштабу классифицируются на локальные, групповые, корпоративные.

Локальные информационные системы реализуются, как правило, на одном компьютере. В современных браузерах есть встроенные средства, позволяющие создавать и использовать такого рода информационные системы. К плюсам можно отнести независимость от работы глобальной и локальной сетей. Однако если возникает необходимость работать с базой данных нескольким пользователям с разных компьютеров, то согласовывать внесенные изменения не получается.

Групповые информационные системы ориентированы на коллективное использование информации членами рабочей группы и чаще всего строятся на базе локальной вычислительной сети.

Корпоративные информационные системы являются развитием систем для рабочих групп, они ориентированы на крупные компании и могут поддерживать территориально разнесённые узлы и сети.

По способу организации классифицируются на следующие: системы на основе архитектуры файл-сервер, системы на основе архитектуры клиент-сервер, системы на основе многоуровневой архитектуры, системы на основе Интернет-технологий.

Для работы с *системами на основе архитектуры файл-сервер* существуют специальные системы управления базами

данных, например, *MS Access*. Они работают на компьютерах пользователей. Сервер не участвует в обработке данных, а только хранит их.

При большом количестве пользователей возникают следующие недостатки файл-серверных информационных систем:

– слабая защита от несанкционированного доступа;

– компьютеры пользователей должны быть достаточно мощными, чтобы успешно обрабатывать информацию;

– ненадежность при внесении изменений в базу данных.

Для устранения перечисленных недостатков СУБД помещают на сервер.

Система на основе архитектуры клиент-сервер находится на том же компьютере, где и сама база данных, беря на себя всю работу с данными. Для этого чаще всего используется язык структурных запросов *SQL* (*Structured Query Language*). Серверная и клиентская части могут быть установлены на одном компьютере.

Преимущества клиент-серверных СУБД:

– надежная защита данных и работа при большом количестве пользователей;

– компьютеры пользователей не обязательно должны быть мощными, так как вся обработка данных выполняется сервером;

– для модернизации СУБД достаточно работать с сервером.

Недостатки: высокая стоимость СУБД Oracle, *MS SQL Server*.

Системы на основе многоуровневой архитектуры. Под архитектурным уровнем СУБД понимают функциональный компонент, механизмы которого служат для поддержки некоторого уровня абстракции данных (логический, физический, внешний уровень). По числу уровней в архитектуре различают одноуровневые, двухуровневые и трехуровневые системы (рис. 3.2).

Многоуровневая архитектура СУБД, ставшая следствием развития архитектуры клиент-сервер, состоит из трех уровней:

– нижний уровень представляет собой приложения клиентов;

– средний уровень – это сервер приложений;

– верхний уровень – это удаленный специализированный сервер базы данных.

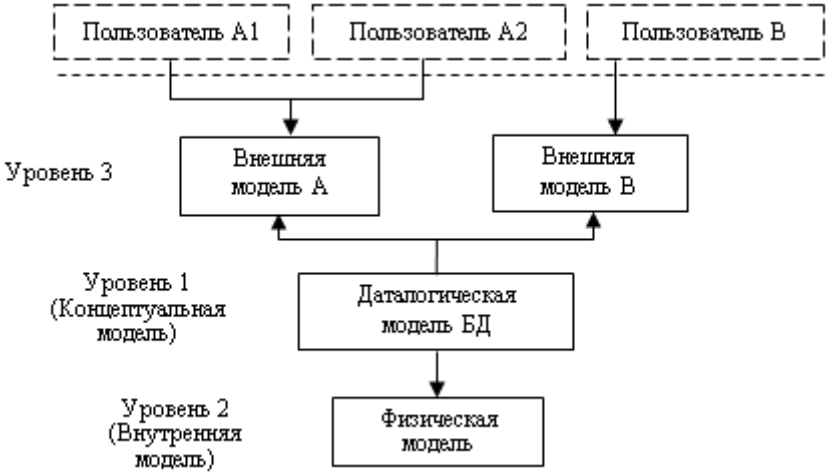


Рис. 3.2. Классификация СУБД по числу уровней в архитектуре

Практически все современные системы управления базами данных ориентированы на работу с реляционной моделью данных. Реляционную модель можно представить как особый метод рассмотрения данных, который включает в себя и сами данные, и способы работы и манипуляции с ними. Другими словами, в реляционных БД используются таблицы, между которыми устанавливаются связи, и информация, введенная в одну таблицу, может быть связана с одной или несколькими записями из другой таблицы. Важным понятием в теории реляционных БД является нормализация.

Основные принципы нормализации:

Каждая таблица состоит из однотипных строк и имеет уникальное имя.

В каждой позиции таблицы на пересечении строки и столбца находится всегда только одно значение.

В таблице не может быть одинаковых строк.

Столбцы имеют разные имена и содержат однородные значения данных (фамилии, даты, денежные суммы и т.д.).

В каждой таблице должен быть первичный ключ (ключевой столбец), который однозначно определяет любую запись.

Задание

В табл.3.1 отражены фамилии студентов, занимающихся научной работой по теме «Моделирование транспортных процессов», в табл. 3.2 – фамилии студентов, названия посещаемых ими секций.

Таблица 3.1 БД студентов, занимающихся научной работой

Номер п/п	Студент	Тема
1	Денисова М.Ю.	Моделирование
2	Алейникова А.О.	Моделирование
3	Соломейчук А.Н.	Проектирование
4	Пиховкин Е.Т.	Моделирование
5	Юрков С.Ю.	Проектирование
6	Сапожников А.Л.	Моделирование
7	Крейнин А.Б.	Проектирование

Таблица 3.2 БД секций по языкам программирования

Номер п\п	Студент	Язык программирования
1	Денисова М.Ю.	Паскаль
2	Алейникова А.О.	С++
3	Соломейчук А.Н.	Паскаль
4	Пиховкин Е.Т.	
5	Юрков С.Ю.	Паскаль
6	Сапожников А.Л.	С++
7	Крейнин А.Б.	С++

Необходимо определить, сколько студентов, занимающихся моделированием, изучают язык программирования С++.

В *системах на основе Интернет-технологий* основное внимание уделяется разработке инструментальных программных средств. Структура информационного приложения имеет следующий вид: браузер – сервер приложений – сервер баз данных – сервер динамических страниц – web-сервер.

По сфере применения СУБД делятся на четыре типа (рис. 3.3):

1. Информационно-справочные системы.
2. Системы обработки транзакций.
3. Системы поддержки принятия решений.
4. Офисные информационные системы.

Рис. 3.3. Классификация информационных систем по сфере применения

Информационные системы также можно условно разделить на *фактографические*, *документальные* и *экспертные*.

В фактографических ИС регистрируются факты – конкретные значения данных об объектах реального мира. Основная идея таких систем заключается в том, что все сведения об объектах (фамилии людей и названия предметов, числа, даты) вводятся в заранее обусловленном формате. Информация, с которой работает фактографическая ИС имеет четкую структуру, позволяющую машине отличать одно данное от другого, поэтому фактографическая ИС способна давать однозначные ответы на запросы. Например, «Какие культурно-исторические памятники Санкт-Петербурга включены в список ЮНЕСКО?», «Кто из сотрудников фирмы имеет двоих детей?» и т.д.

Документальные ИС работают с принципиально другим классом задач, которые не предполагают однозначного ответа на поставленный вопрос. Основу таких систем составляет совокупность неструктурированных текстовых документов (статьи, книги, рефераты, тексты законов) и графических объектов, снабженная формальным аппаратом поиска. Цель системы – выдать в ответ на запрос список документов или объектов, в какой-то мере удовлетворяющих сформулированным в запросе условиям. Документальная система должна по контексту определять смысл того или иного термина. Например, различать «Близнецы» (созвездие) и «Близнецы» (люди).

Следует отметить, что современные фактографические

системы часто работают с неструктурированными блоками информации (текстами, графикой, звуком, видео), снабженными структурированными описателями.

Экспертные системы (ЭС) – интеллектуальные системы, призванные играть роль «советчика», построены на базе формализованного опыта и знаний эксперта. Ядром ЭС являются базы знаний, в которых собраны знания экспертов (специалистов) в определенной области, на основе которых ЭС позволяет моделировать рассуждения специалистов из данной предметной области.

Требования, предъявляемые к информационным системам: гибкость, надежность, эффективность и безопасность.

Гибкость – способность к адаптации и дальнейшему развитию подразумевают возможность приспособления информационной системы к новым условиям, новым потребностям предприятия.

Надежность. Требование надежности обеспечивается созданием резервных копий хранимой информации, выполнения операций протоколирования, поддержанием качества каналов связи и физических носителей информации, использованием современных программных и аппаратных средств.

Эффективность. Информационная система является эффективной, если с учётом выделенных ей ресурсов она позволяет решать возложенные на неё задачи в минимальные сроки. Эффективность системы обеспечивается оптимизацией данных и методов их обработки, применением оригинальных разработок, идей, методов проектирования.

Безопасность. Требование безопасности обеспечивается современными средствами разработки информационных систем, современной аппаратурой, методами защиты информации, применением паролей и протоколированием, постоянным мониторингом состояния безопасности операционных систем и средств их защиты.

3.2. Модели данных

3.2.1. Иерархическая модель

Иерархическая структура представляет совокупность элементов, связанных между собой по определенным правилам. Графическим способом представления иерархической структуры является дерево (рис.3.4).

Дерево представляет собой иерархию элементов,

называемых узлами. Под элементами понимается совокупность атрибутов, описывающих объекты. В модели имеется корневой узел (корень дерева), который находится на самом верхнем уровне и не имеет узлов, стоящих выше него. У одного дерева может быть только один корень. Остальные узлы, называемые порожденными, связаны между собой следующим образом: каждый узел имеет только один исходный, находящийся на более высоком уровне, и любое число (один, два или более, либо ни одного) подчиненных узлов на следующем уровне

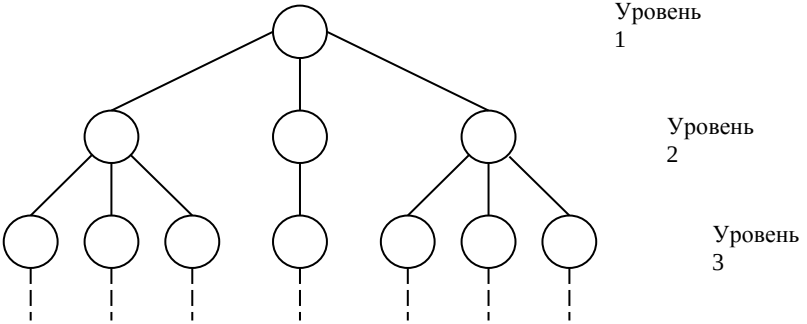


Рис. 3.4. Графическое изображение иерархической структуры

Примером простого иерархического представления может служить административная структура высшего учебного заведения: институт – отделение – факультет – студенческая группа (рис. 3.5).

К *достоинствам* иерархической модели данных относятся эффективное использование памяти ЭВМ и неплохие показатели времени выполнения операций над данными.

Недостатком иерархической модели является ее громоздкость для обработки информации с достаточно сложными логическими связями.

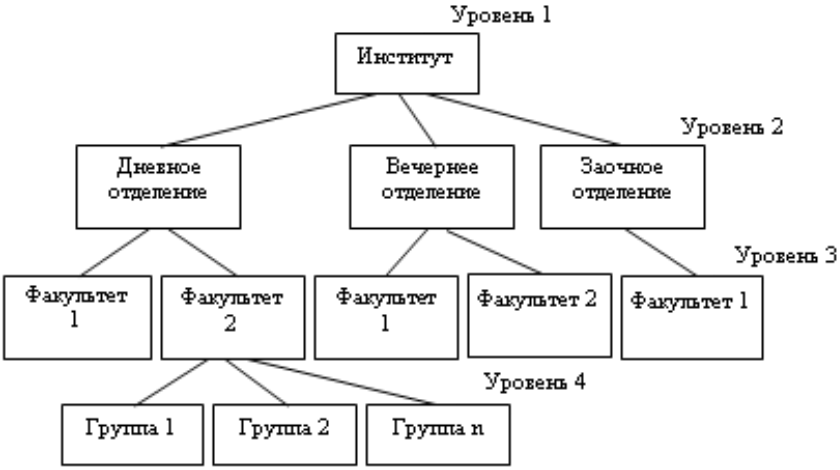


Рис. 3.5. Пример иерархической структуры

На иерархической модели данных основано сравнительно ограниченное количество СУБД, в числе которых можно назвать зарубежные системы *IMS*, *PC/Focus*, *Team-Up* и *Data Edge*, а также отечественные системы Ока, ИНЭС и МИРИС.

3.2.2. Сетевая модель данных

Отличие сетевой структуры от иерархической заключается в том, что каждый элемент в сетевой структуре может быть связан с любым другим элементом (рис. 3.6). Пример простой сетевой структуры показан на рис. 3.7.

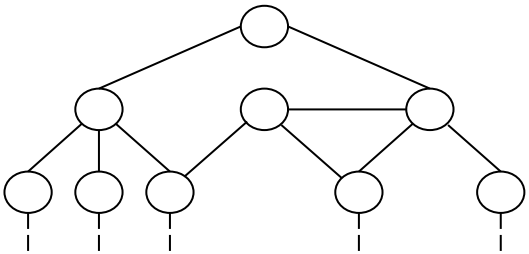


Рис. 3.6. Графическое изображение сетевой структуры

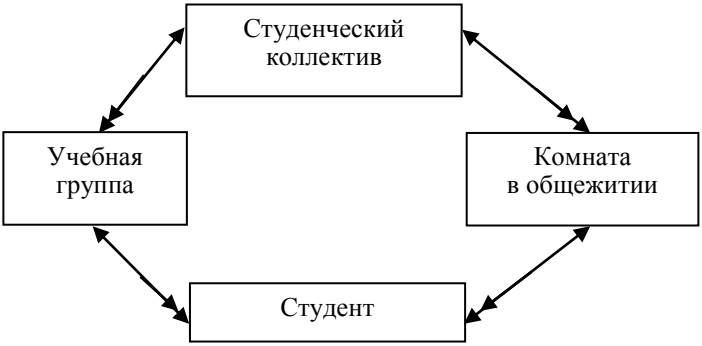


Рис. 3.7. Пример взаимосвязей между элементами сетевой структуры

Достоинством сетевой модели данных является возможность эффективной реализации по показателям затрат памяти и оперативности.

Недостатком сетевой модели данных являются высокая сложность и жесткость схемы БД, построенной на ее основе.

Наиболее известными сетевыми СУБД являются IDMS, db_VistaIII, СЕТЬ, СЕТОР и КОМПАС.

3.2.3. Реляционная модель данных

Реляционная база данных представляет собой хранилище данных, организованных в виде двумерных таблиц. Любая таблица реляционной базы данных состоит из строк (называемых также записями) и столбцов (называемых также полями).

Строки таблицы содержат сведения о представленных в ней фактах (или документах, или людях, одним словом, – об однотипных объектах). На пересечении столбца и строки находятся конкретные значения содержащихся в таблице данных.

Данные в таблицах удовлетворяют следующим принципам:

1. Каждое значение, содержащееся на пересечении строки и столбца, должно быть атомарным.
2. Значения данных в одном и том же столбце должны принадлежать к одному и тому же типу, доступному для использования в данной СУБД.
3. Каждая запись в таблице уникальна, то есть в таблице не существует двух записей с полностью совпадающим набором значений ее полей.
4. Каждое поле имеет уникальное имя.
5. Последовательность полей в таблице несущественна.

6. Последовательность записей в таблице несущественна.

Несмотря на то, что строки таблиц считаются неупорядоченными, любая система управления базами данных позволяет сортировать строки и столбцы в выборках из нее нужным пользователю способом.

Поскольку последовательность полей в таблице несущественна, обращение к ним производится по имени, и эти имена для данной таблицы уникальны (но не обязаны быть уникальными для всей базы данных).

Поле или комбинацию полей, значения которых однозначно идентифицируют каждую запись таблицы, называют ***возможным ключом*** (или просто ***ключом***).

Если таблица имеет более одного возможного ключа, тогда один ключ выделяют в качестве *первичного*. Первичный ключ любой таблицы обязан содержать уникальные непустые значения для каждой строки.

Поле, указывающее на запись в другой таблице, связанную с данной записью, называется *внешним ключом*. Иначе говоря, внешний ключ – это поле или набор полей, чьи значения совпадают с имеющимися значениями первичного ключа другой таблицы.

Подобное взаимоотношение между таблицами называется ***связью***. Связь между двумя таблицами устанавливается путем присвоения значений внешнего ключа одной таблицы значениям первичного ключа другой.

Группа связанных таблиц называется ***схемой базы данных***. Информация о таблицах, их полях, первичных и внешних ключах, а также иных объектах базы данных, называется ***метаданными***.

Достоинство реляционной модели данных заключается в простоте, понятности и удобстве физической реализации на ЭВМ. Именно простота и понятность для пользователя явились основной причиной ее широкого использования.

3.2.4. Постреляционная модель

Классическая реляционная модель предполагает неделимость данных, хранящихся в полях записей таблиц. Постреляционная модель представляет собой расширенную реляционную модель, снимающую ограничение неделимости данных. Модель допускает многозначные поля – поля, значения которых состоят из подзначений. Набор значений многозначных полей считается самостоятельной таблицей, встроенной в

основную таблицу.

На рис. 3.8 на примере информации о накладных и товарах для сравнения дано представление одних и тех же данных с помощью реляционной и постреляционной моделей.

Поскольку постреляционная модель допускает хранение в таблицах ненормализованных данных, возникает проблема обеспечения целостности и непротиворечивости данных. Эта проблема решается включением в СУБД соответствующих механизмов.

Накладные

N накладной	Покупатель
10773	17723
8374	8232
7353	8723

Накладные-товары

N накладной	Товар	Количество
10773	Шампунь-полироль	5
0373	Тормозная жидкость	2
8374	Антифриз	4
8374	Салонные фильтры	6
8374	Автопарфюмерия	2
7364	Автохимия	1

а)

Накладные

N накладной	Покупатель	Товар	Количество
0373	17723	Шампунь-полироль	5
		Тормозная жидкость	2
8374	8232	Антифриз	4
		Салонные фильтры	6
		Автопарфюмерия	2
7364	8723	Автохимия	1

б)

Рис. 3.8. Структуры данных реляционной (а) и постреляционной (б) моделей

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.

Рассмотренная постреляционная модель данных

поддерживается СУБД *uniVers*. К числу других СУБД, основанных на постреляционной модели данных, относятся также системы *Bubba* и *Dasdb*.

3.2.5. Многомерная модель

Многомерный подход к представлению данных появился практически одновременно с реляционным, но интерес к многомерным СУБД стал приобретать массовый характер с середины 90-х годов. Толчком послужила в 1993 году статья Э. Кодда. В ней были сформулированы 12 основных требований к системам класса *OLAP* (*OnLine Analytical Processing* – оперативная аналитическая обработка), важнейшие из которых связаны с возможностями концептуального представления и обработки многомерных данных.

В развитии концепций информационных систем можно выделить следующие два направления:

1) системы оперативной (транзакционной) обработки;

2) системы аналитической обработки (системы поддержки принятия решений).

Реляционные СУБД предназначались для информационных систем оперативной обработки информации и в этой области весьма эффективны. В системах аналитической обработки они показали себя несколько неповоротливыми и недостаточно гибкими. Более эффективными здесь оказываются многомерные СУБД.

Многомерные СУБД являются узкоспециализированными СУБД, предназначенными для интерактивной аналитической обработки информации. Основные понятия, используемые в этих СУБД: агрегируемость, историчность и прогнозируемость.

Агрегируемость данных означает рассмотрение информации на различных уровнях ее обобщения. В информационных системах степень детальности представления информации для пользователя зависит от его уровня: аналитик, пользователь, управляющий, руководитель.

Историчность данных предполагает обеспечение высокого уровня статичности собственно данных и их взаимосвязей, а также обязательность привязки данных ко времени.

Прогнозируемость данных подразумевает задание функций прогнозирования и применение их к различным временным интервалам.

Многомерность модели данных означает не многомерность визуализации цифровых данных, а многомерное логическое

представление структуры информации при описании и в операциях манипулирования данными.

По сравнению с реляционной моделью многомерная организация данных обладает более высокой наглядностью и информативностью (рис. 3.9).

Модель	Месяц	Объем
Мерседес	сентябрь	11
Мерседес	октябрь	14
Мерседес	ноябрь	5
Рено логан	сентябрь	3
Рено логан	сентябрь	18
Ауди	октябрь	19

а)

Модель	Сентябрь	Октябрь	Ноябрь
Мерседес	11	24	5
Рено Логан	2	18	No
Ауди	No	19	No

б)

Рис. 3.9. Реляционное (а) и многомерное (б) представление данных об объемах продаж автомобилей

Если речь идет о многомерной модели с мерностью больше двух, то не обязательно визуально информация представляется в виде многомерных объектов (трех-, четырех- и более мерных гиперкубов). Пользователю и в этих случаях более удобно иметь дело с двумерными таблицами или графиками. Данные при этом представляют собой «вырезки» из многомерного хранилища данных, выполненные с разной степенью детализации.

Основные понятия многомерных моделей данных: измерение и ячейка.

Измерение – это множество однотипных данных, образующих одну из граней гиперкуба. В многомерной модели измерения играют роль индексов, служащих для идентификации конкретных значений в ячейках гиперкуба.

Ячейка – это поле, значение которого однозначно определяется фиксированным набором измерений. Тип поля чаще всего определен как цифровой. В зависимости от того, как формируются значения некоторой ячейки, она может быть переменной (значения изменяются и могут быть загружены из внешнего источника данных или сформированы программно) либо формулой (значения, подобно формульным ячейкам электронных таблиц, вычисляются по заранее заданным формулам).

На рис. 3.9, *б* каждое значение ячейки *Объем продаж* однозначно определяется комбинацией временного измерения *Месяц продаж* и модели автомобиля. На практике зачастую

требуется большее количество измерений. Пример трехмерной модели данных представлен на рис. 3.10.

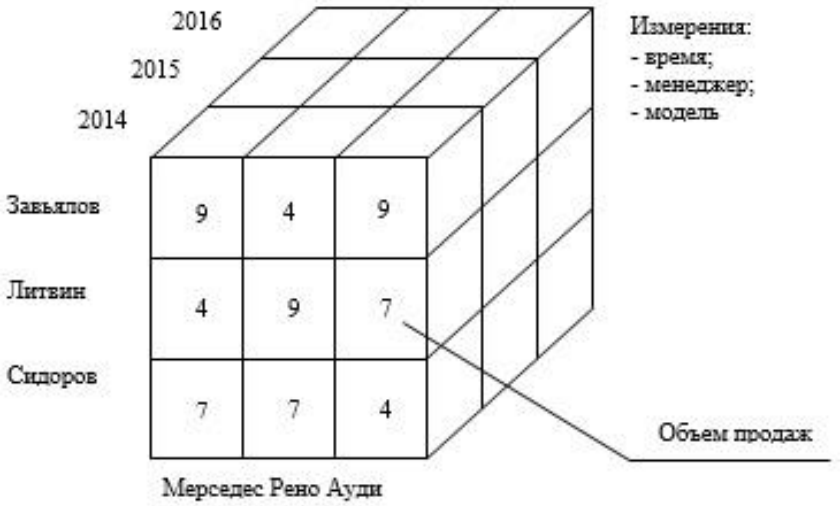


Рис. 3.10. Пример трехмерной модели

В существующих многомерных СУБД используются две основные схемы организации данных: гиперкубическая и поликубическая.

В *поликубической* схеме предполагается, что в БД может быть определено несколько гиперкубов с различной размерностью и с различными измерениями в качестве граней. Примером системы, поддерживающей поликубический вариант БД, является сервер *Oracle Express Server*.

В случае *гиперкубической* схемы предполагается, что все ячейки определяются одним и тем же набором измерений. Это означает, что при наличии нескольких гиперкубов в БД, все они имеют одинаковую размерность и совпадающие измерения.

Основным *достоинством* многомерной модели данных является удобство и эффективность аналитической обработки больших объемов данных, связанных со временем.

Недостатком многомерной модели данных является ее громоздкость для простейших задач обычной оперативной обработки информации.

Примерами систем, поддерживающими многомерные модели данных, является *Essbase*, *Media Multi-matrix*, *Oracle*

Express Server, *Cache*. Существуют программные продукты, например *Media/MR*, позволяющие одновременно работать с многомерными и с реляционными БД.

3.2.6. Объектно-ориентированная модель

В объектно-ориентированной модели при представлении данных имеется возможность идентифицировать отдельные записи базы данных. Между записями и функциями их обработки устанавливаются взаимосвязи с помощью механизмов, подобных соответствующим средствам в объектно-ориентиро-ванных языках программирования.

Стандартизированная объектно-ориентированная модель описана в рекомендациях стандарта *ODMG*-93 (Object Database Management *Group* – группа управления объектно-ориентированными базами данных).

Рассмотрим упрощенную модель объектно-ориенти-рованной БД. Структура объектно-ориентированной БД графически представима в виде дерева, узлами которого являются объекты. Свойства объектов описываются некоторым стандартным типом или типом, конструируемым пользователем (определяется как *class*). Значение свойства типа *class* есть объект, являющийся экземпляром соответствующего класса. Каждый объект-экземпляр класса считается потомком объекта, в котором он определен как свойство. Объект-экземпляр класса принадлежит своему классу и имеет одного родителя. Родовые отношения в БД образуют связную иерархию объектов. Пример логической структуры объектно-ориентированной БД библиотечного дела дан на рис. 3.11. Здесь объект типа *Библиотека* является родительским для объектов-экземпляров классов *Абонент*, *Каталог* и *Выдача*. Различные объекты типа *Книг*а могут иметь одного или разных родителей. Объекты типа *Книга*, имеющие одного и того же родителя, должны различаться, по крайней мере, инвентарным номером (уникален для каждого экземпляра книги), но имеют одинаковые значения свойств *isbn*, *удк*, *название* и *автор*.

Логическая структура объектно-ориентированной БД внешне похожа на структуру иерархической БД. Основное различие между ними состоит в методах манипулирования данными.

Для выполнения действий над данными в рассматриваемой модели БД применяются логические операции, усиленные объектно-ориентированными механизмами инкапсуляции,

наследования и полиморфизма.

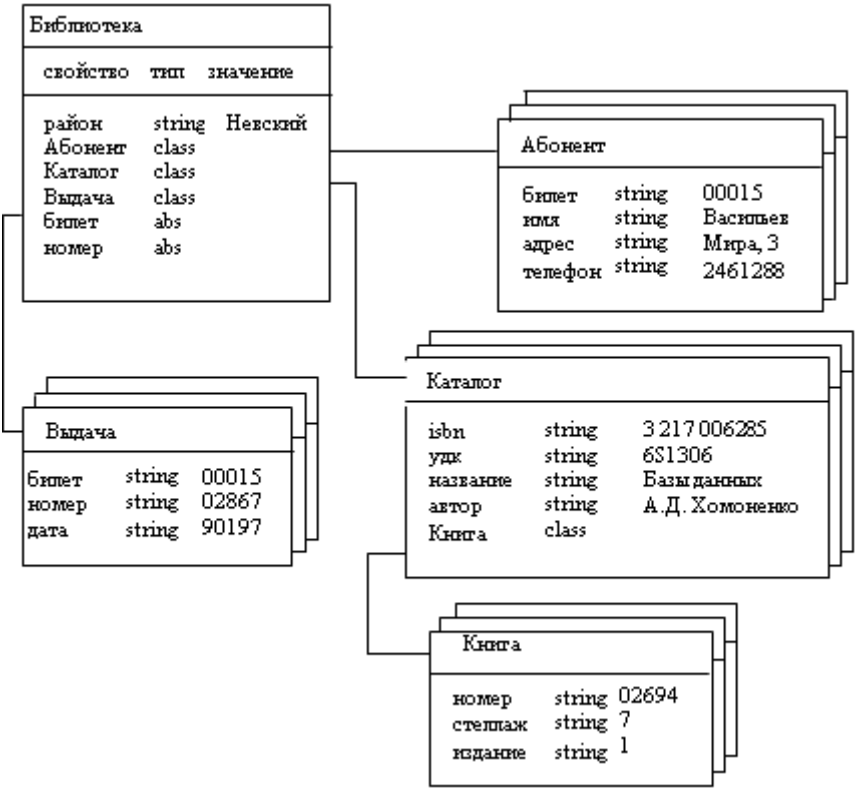


Рис. 3.11. Логическая структура объектно-ориентированной БД

Инкапсуляция ограничивает область видимости имени свойства пределами того объекта, в котором оно определено. Если в объект типа *Каталог* добавить свойство, задающее телефон автора книги и имеющее название *телефон*, то мы получим одноименные свойства у объектов *Абонент* и *Каталог*. Смысл такого свойства будет определяться тем объектом, в который оно инкапсулировано. *Наследование* распространяет область видимости свойства на всех потомков объекта. Всем объектам типа *Книга*, являющимся потомками объекта типа *Каталог*, можно приписать свойства объекта-родителя: *isbn*, *удк*, *название* и *автор*. Если необходимо расширить действие механизма наследования на объекты, не являющиеся

непосредственными родственниками (например, между двумя потомками одного родителя), то в их общем предке определяется абстрактное свойство типа *abs*. Так, определение абстрактных свойств *билет* и *номер* в объекте *Библиотека* приводит к наследованию этих свойств всеми дочерними объектами *Абонент*, *Книга* и *Выдача*. *Полиморфизм* означает способность одного и того же программного кода работать с разнотипными данными. Одни и те же методы оперируют с разными объектами в зависимости от типа аргумента. В данном примере полиморфизм означает, что объекты класса *Книга*, имеющие разных родителей из класса *Каталог*, могут иметь разный набор свойств. Программы работы с объектами класса *Книга* могут содержать полиморфный код. Поиск в БД состоит в выяснении сходства между объектом, задаваемым пользователем, и хранящимися объектами.

4. СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ MS ACCESS

MS Access – это система управления реляционными базами данных, предназначенная для работы на автономном ПК, в локальной сети под управлением операционной системы Windows или в Internet.

MS Access содержит набор средств для поддержки таблиц и отношений между ними. При обработке данных в *MS Access* используется структурированный язык запросов *SQL*, который можно назвать стандартным языком базы данных. С его помощью выполняется самая разнообразная обработка данных:

необходимая выборка данных;
внесение изменений в базу данных;
преобразование и удаление таблицы;
выполнение различных расчетов и др.

К основным объектам относятся: таблица, форма, запрос, отчет, макрос.

Таблица – это объект, предназначенный для хранения данных в виде записей (строк) и полей (столбцов). Обычно каждая таблица используется для хранения сведений по одному конкретному вопросу.

Форма – объект, предназначенный, в основном, для ввода данных. В форме можно разместить элементы управления, применяемые для ввода, изображения и изменения данных в

полях таблиц.

Запрос – это объект, позволяющий получить нужные данные из одной или нескольких таблиц.

Отчет – объект, предназначенный для печати данных.

Макрос – это средство для автоматизации часто выполняемых операций. Макрос содержит одну или несколько макрокоманд.

Начинать следует с создания таблицы.

В таблице сохраняют записи, содержащие сведения определенного типа, например, список клиентов или опись товаров. Составной частью таблицы являются поля.

Поле – это элемент таблицы, который содержит данные определенного рода, например, фамилию сотрудника. В режиме таблицы для представления поля используется столбец или ячейка, в этом случае имя поля является заголовком столбца таблицы.

Запись – это полный набор данных об определенном объекте. В режиме таблицы запись изображается как строка.

При создании таблицы необходимо указывать типы данных. Рассмотрим основные типы данных:

Числовой. Используется для полей, со значениями которых проводятся арифметические операции.

Текстовый. Текстовыми данными представляют фамилии людей, названия фирм, организаций и т.д. Значение каждого текстового данного представлено последовательностью символов и цифр длиною не более 255.

Дата/Время. Используется для задания дат и времени в определенном формате. Например, задание даты в формате ДД.ММ.ГГ (день, месяц, год).

Счетчик. Используется для автоматической нумерации записей.

Поле МЕМО. Используется для хранения длинного текста или комбинации текста и чисел.

Логический. Содержит логические величины.

Возможны также следующие типы данных:

Поле объекта OLE.

Гиперссылка.

Вложение.

Мастер подстановок.

В процессе создания таблицы может использоваться маска ввода. Маска ввода – это шаблон, позволяющий вводить в поле значения, имеющие одинаковый формат. Маска ввода

автоматически изображает в поле постоянные символы и достаточно заполнить пустые позиции. Ввести данные, не вписывающиеся в маску ввода нельзя.

После того как определены все поля, необходимо указать первичный ключ таблицы. ***Первичный ключ таблицы*** – это одно или несколько полей, совокупность значений которых однозначно определяет любую запись в таблице.

Любая таблица может быть представлена в двух режимах:

Режим таблицы, предназначенный для работы с записями (ввод и редактирование данных).

Режим конструктора, предназначенный для работы с полями.

В СУБД *MS Access* любой объект (таблицу, форму, запрос, отчет) можно создавать вручную либо с помощью Мастера.

Рассмотрим создание таблицы вручную.

1. Запускаем приложение *MS Access*.
2. Выбираем вкладку «Новая база данных» и указываем полное имя файла, в котором будет храниться база данных (рис. 4.1).

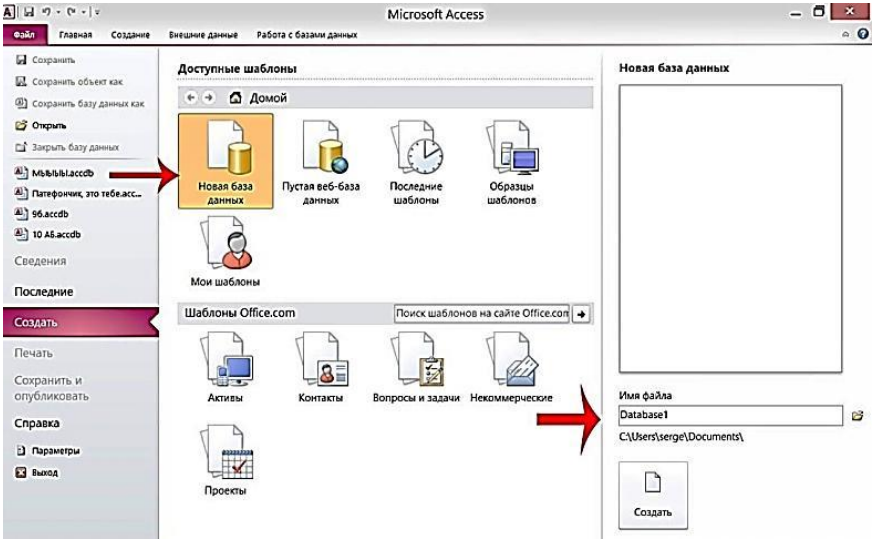


Рис. 4.1. Создание новой базы данных

3. Нажимаем кнопку «Создать».
4. На этом этапе создана новая база данных. Приступаем к формированию таблицы с помощью Конструктора (рис. 4.2).

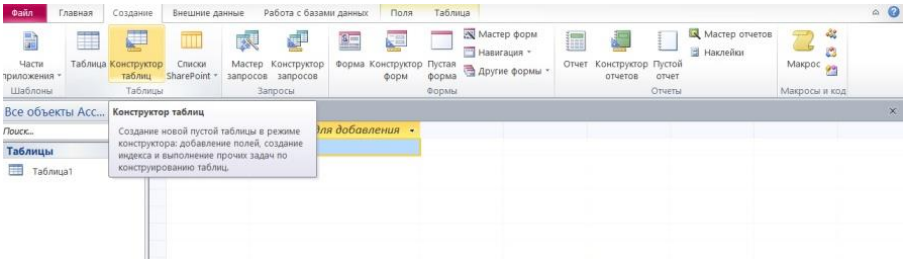


Рис. 4.2. Режим «Конструктор»

Появляется окно с макетом создаваемой таблицы и бланком свойств. В макете указываются поля и типы данных. В бланке свойств можно указывать значение поля по умолчанию, маску ввода и ряд других свойств (рис. 4.3).

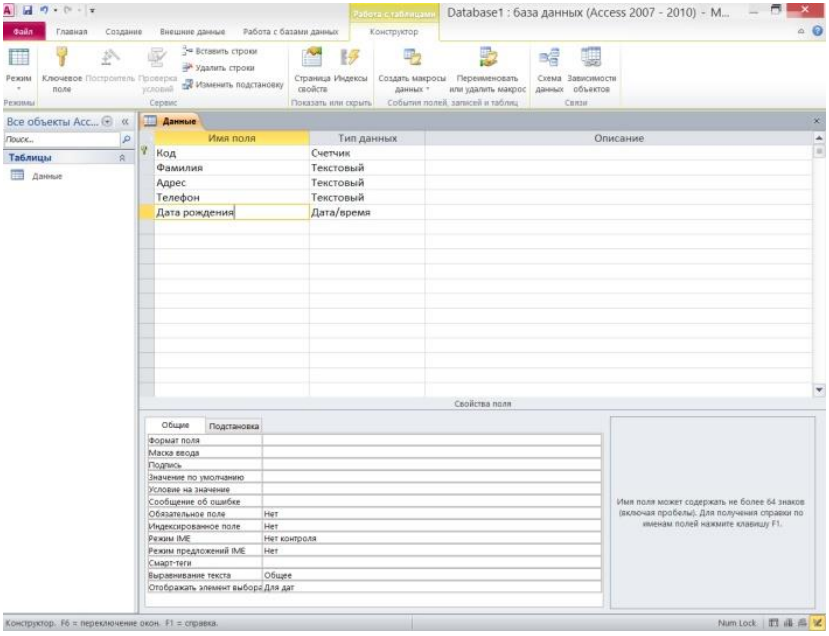


Рис. 4.3. Создание таблицы в режиме конструктора

5. Завершив создание макета таблицы, необходимо его сохранить и перейти в режим таблицы.

6. В режиме таблицы можно заносить и корректировать

данные (рис. 4.4).

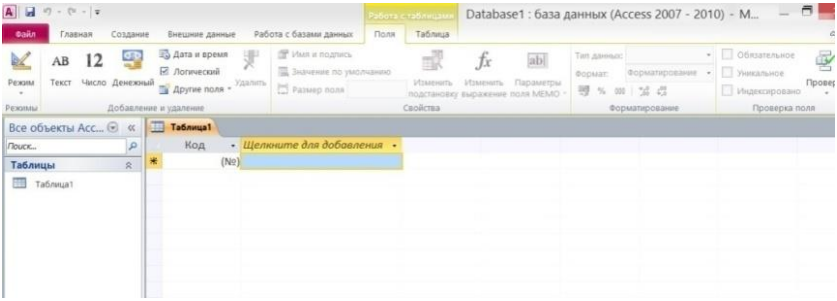


Рис. 4.4. Заполнение базы данных в режиме «Таблица»

Запросы. В общем случае запрос – это объект *MS Access*, предназначенный для получения данных из одной или нескольких таблиц.

Существуют различные типы запросов, но мы изучим создание простейшего типа запросов: запрос-выборку.

Запрос-выборка – это производная таблица, которая содержит те же структурные элементы, что и обычная таблица, и формируется на основе фактических данных системы. При создании макета запроса в общем случае нужно выполнить следующие операции:

1. Указать системе, какие поля и из каких таблиц включить в запрос.
2. Описать групповые операции.
3. Указать условие отбора.

При разработке конкретного запроса допускается любое сочетание вышеуказанных операций.

Обратите внимание, что запрос – это единственный объект, для которого можно указать несколько таблиц и (или) запросов.

Создание запроса:

1. В окне базы данных выбираем вкладку «Создание», затем группу «Запросы», а в ней – «Конструктор запросов» (рис. 4.5).

2. На переднем плане появится окно диалога «Добавление таблицы».

3. Выделяем необходимую таблицу и нажимаем кнопку *Добавить.*

4. Закрываем окно диалога «Добавление таблицы»
5. Активизируем окно «Запрос-выборка».
6. В первую клетку строки *Поле* бланка запроса

перетаскиваем из добавленной таблицы необходимые для выборки данных поля.

7. При необходимости указываем условия отбора и сортировки.

8. Выбираем команду «Выполнить» для просмотра полученных данных.

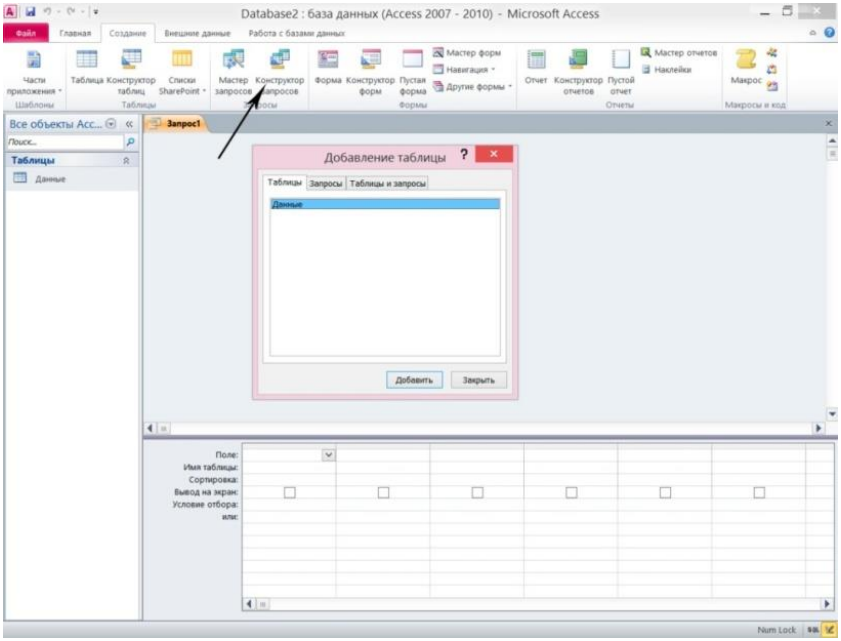


Рис. 4.5. Создание запроса

Данные в запросе можно сортировать по убыванию или по возрастанию. Для этого в режиме «Конструктор запросов» в строке *Сортировка* щелкните мышью под полем, по которому хотите произвести сортировку и укажите ее вид.

Для задания в запросе условия нужно воспользоваться строкой *Условие отбора* и строкой *Или.*

Условия в запросах могут быть различными. Можно указывать конкретные значения полей. В строке *Или* можно указать второе возможное значение поля.

В выражениях для условий отбора допускается использование шаблонов (табл. 4.1):

1. *Звездочка (*)* – заменяет любую группу любых символов.
2. *Знак вопроса (?)* – заменяет любой один символ.
3. *Знак (#)*-заменяет одну цифру.

4. Квадратные скобки [] обозначают выбор одного символа из указанного набора.

5. *Восклицательный знак (!) в сочетании с квадратными скобками []* – исключение символов, указанных в скобках после восклицательного знака.

Таблица 4.1 Примеры использования операторов и шаблонов в запросах

Пример	Описание
Like «* а»	Содержимое текстового поля оканчивается на "а"
Like «*ов»or «*ова»	Содержимое текстового поля оканчивается на «ов» или «ова»
Between «А*» and «Н*»	Первая буква текстового поля находится «в диапазоне от А до Н
Between 1 and 10	Содержимое числового поля находится в диапазоне от 1 до 10
= 100	Содержимое числового поля равно 100
Петров[а,ы]	Содержимое поля равно «Петрова» или «Петровы»
Петров[!а, ы]	Содержимое поля не может быть «Петрова» или «Петровы», но возможно значение, например, "Петрову"

Операторы *between and* используются при задании диапазона значений поля.

Операторы *And, Or* означают соответственно *И, Или.*

Создание запроса с параметром. Запрос с параметром позволяет определить одно или несколько условий отбора во время выполнения запроса.

Запрос с параметром создается точно так же, как и запрос-выборка, но в поле, содержащем нужные записи, в строке условия отбора в квадратных скобках записывается критерий отбора. Например: [Укажите фамилию].

Создание запроса с добавлением вычисляемого поля. Вычисляемые поля в запросы добавляются с целью выполнения расчетов в базе данных. Рассмотрим алгоритм создания вычисляемых полей в уже существующем запросе-выборке:

1. Открыть запрос в режиме «Конструктор».
2. В первом пустом столбце бланка запроса в первой строке нажать правую кнопку мыши и в появившемся контекстном

меню выбрать пункт «Построить» (рис. 4.6).

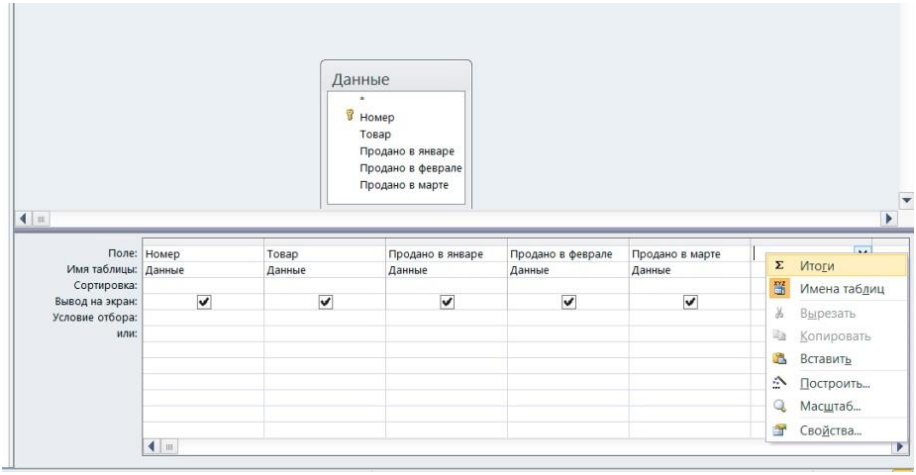


Рис. 4.6. Создание вычисляемых полей. Вызов Построителя

3. В появившемся диалоговом окне в бланке Построитель записываем имя добавляемого поля. Ставим знаки «:» и «=».

4. В левой нижней части Построителя открываем папку Запросы, выбираем запрос, в который добавляется вычисляемое поле.

5. В средней нижней части бланка выдается список допустимых полей. Выбираем двойным щелчком мыши поле, участвующее в вычислении, и заносим его в верхнюю строку после знака «=», записываем знак арифметической операции.

6. Продолжаем выбор полей и расстановку знаков операций (рис. 4.7).

7. По окончании обязательно нажимаем на кнопку «ОК» и сохраняем изменения в запросе.

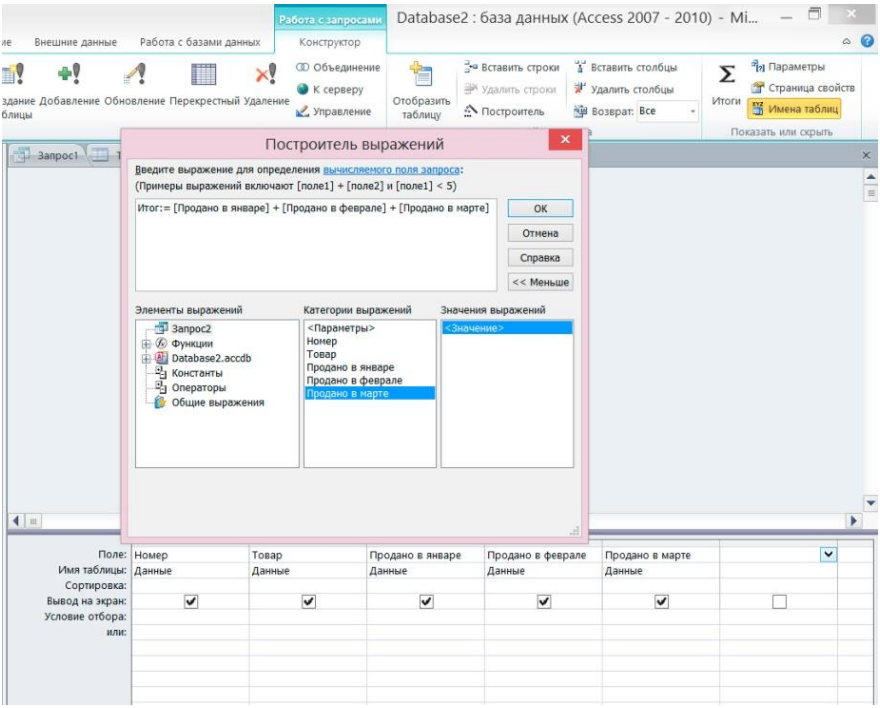


Рис. 4.7. Создание вычисляемых полей. Работа в Вычислителе

С другой стороны, задача преобразования полученного описания предметной области в схему базы данных тоже может оказаться очень непростой. Уже сам выбор СУБД может оказаться сложным делом. Сегодня существует достаточно много СУБД, каждая из которых обладает достоинствами и недостатками. Система, которая годилась бы для любой задачи, пока не создано и вряд ли она когда-нибудь появится. Существенное значение имеет и экономический аспект – промышленные СУБД могут предоставлять богатый набор возможностей, но и стоимость у них окажется соответствующей.

Выбор системы управления базами данных осуществляется с учетом потребностей пользователя одним из следующих методов: анализа возможностей, экспериментальной проверки, имитации и моделирования. От правильного выбора инструментальных средств создания информационных систем, определения подходящей модели данных, обоснования рациональной схемы построения базы данных, организации запросов к хранимым данным и ряда других моментов во многом зависит эффективность функционирования

разрабатываемых систем. Все это требует осознанного применения теоретических положений и инструментальных средств разработки баз данных и информационных систем

Метод анализа возможностей основан на балльной оценке приведенных выше характеристик СУБД с точки зрения требований пользователя. Каждая характеристика изучается с двух позиций – присутствует ли она в предлагаемой СУБД и каково ее качество. Качество ранжируется по стандартной шкале. Коэффициент ранжирования умножается на выделенный для данной составляющей вес, и взвешенные по каждой составляющей баллы суммируются.

Метод экспериментальной проверки состоит в создании определенной прикладной среды и получении с ее помощью эксплуатационных характеристик заданной программно-аппаратной системы. Для экспериментальной проверки необходимо спроектировать и загрузить типовую базу данных; затем с использованием языка манипулирования данными СУБД промоделировать требования по обработке существующих и ожидаемых прикладных программ и выполнить экспериментальную проверку рассматриваемых СУБД.

Основным *достоинством* объектно-ориентированной модели данных в сравнении с реляционной является возможность отображения информации о сложных взаимосвязях объектов. Объектно-ориентированная модель данных позволяет идентифицировать отдельную запись базы данных и определять функции их обработки.

Недостатками объектно-ориентированной модели являются высокая понятийная сложность, неудобство обработки данных и низкая скорость выполнения запросов.

5. ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ МЕТОДОМ «СУЩНОСТЬ – СВЯЗЬ»

Существуют разные подходы к проектированию БД. Рассмотрим два метода проектирования: декомпозиционный и метод «сущность – связь», который рассмотрим первым.

Этапы проектирования

В БД отражается информация об определенной предметной

области. **Предметной областью** называют часть реального мира, представляющую интерес для данного использования. В автоматизированных информационных системах отражение предметной области представляется моделями данных нескольких уровней (число уровней зависит от особенностей СУБД). Независимо от того, поддерживаются ли в явном виде отдельно модели логического и физического уровня, с точки зрения методологии все равно можно выделить эти уровни и соответствующие им этапы проектирования БД.

Первый этап проектирования – инфологическое моделирование. Чтобы спроектировать структуру БД, необходима исходная информация о предметной области. Желательно, чтобы эта информация была представлена в формализованном виде. Описание предметной области, выполненное без ориентации на используемые в дальнейшем программные и технические средства, называется **инфологической моделью** предметной области (ИЛМ).

На втором этапе проектирования на основе инфологической модели строится даталогическая модель БД (ДЛМ). **Даталогическая модель** является моделью логического уровня и представляет собой отображение логических связей между элементами данных безотносительно к их содержанию и среде хранения. Модель строится в терминах информационных единиц, допустимых в той конкретной СУБД, в среде которой проектируется БД. Этап создания ДЛМ называется даталогическим проектированием. Описание логической структуры БД на языке СУБД называется **схемой**.

Третий этап проектирования состоит в привязке ДЛМ к среде хранения с помощью модели данных физического уровня (**физической модели**). Описание физической структуры БД называется **схемой хранения**, соответствующий этап проектирования БД – физическим проектированием.

В ряде СУБД, помимо описания общей логической структуры БД, есть возможность описать логическую структуру БД с точки зрения конкретного пользователя. Такая модель называется **внешней**, а ее описание называется **подсхемой**.

Внешняя модель не всегда является точным подмножеством схемы. Если определена подсхема, то пользователь имеет доступ только к тем данным, которые отражены в соответствующей подсхеме, что является одним из способов защиты информации от несанкционированного доступа.

Рис. 5.1. Взаимосвязь этапов проектирования БД

Взаимосвязь этапов проектирования БД отражена на рис. 3.1, из которого видно, что при проектировании БД возможны возвраты на предыдущие уровни. При этом есть возвраты, обусловленные необходимостью пересмотра результата проектирования, и есть возвраты, вызванные необходимостью уточнения предыдущей модели (обычно инфологической) с целью получения дополнительной информации для проектирования или при выявлении противоречий в модели.

5.1 Инфологическое моделирование

Для описания предметной области инфологического моделирования может использоваться и естественный язык, но его применение имеет много недостатков. Основные: громоздкость описания и неоднозначность трактовки. Поэтому для описания предметной области обычно используют искусственные формализованные языковые средства. В связи со сказанным уточним определение инфологической модели.

Требования, предъявляемые к ИЛМ:

– адекватность отображения предметной области (основное требование);

– ИЛМ должна быть непротиворечивой;

– ИЛМ является конечной, но должна обладать

свойством легкой расширяемости (для обеспечения ввода новых данных без изменения ранее определенных; то же самое можно сказать и об удалении данных);

– язык спецификации ИЛМ должен быть одинаково применим как при ручном, так и при автоматизированном проектировании информационных систем;

– ИЛМ должна легко восприниматься разными категориями пользователей.

Компоненты ИЛМ:

1) описание объектов и связей между ними (ER-модель);
2) описание информационных потребностей пользователей;
3) алгоритмические связи показателей;
4) лингвистические отношения;
5) ограничения целостности.

ER-модель (E (*entity*) – сущность, R (*relationship*) – связь) является центральной компонентой ИЛМ. Вопросы ее построения подробно рассматриваются ниже.

Для описания информационных потребностей пользователей используются специальные языковые средства.

Для отражения алгоритмических связей между показателями используются графы взаимосвязи показателей, отражающие, какие показатели служат исходными для вычисления других (рис. 5.2).

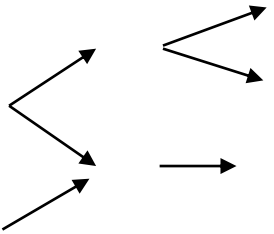


Рис. 5.2. Граф взаимосвязи показателей

Описание предметной области всегда представлено в какой-то знаковой системе. Поэтому кроме отношений, присущих предметной области, возникают еще и отношения, обусловленные особенностями отображения предметной области в языковой среде. Поэтому при построении ИЛМ должны учитываться такие лингвистические категории, как синонимия, омонимия, изоморфизм и др.

Важной компонентой ИЛМ являются ограничения целостности (см. гл. 7).

Построение ER-модели

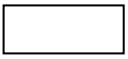
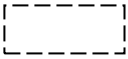
Для описания ИЛМ используются как языки аналитического

(описательного) типа, так и графические средства. Воспользуемся последними как более наглядными.

В предметной области в результате ее анализа выделяют классы объектов. ***Классом объектов*** называют совокупность объектов, обладающих одинаковым набором свойств. Например, если в качестве предметной области рассмотреть высшее учебное заведение, то в данной предметной области можно выделить следующие классы объектов: учащиеся, преподаватели, аудитории, изучаемые дисциплины.

Каждый объект в информационной системе представляется своим идентификатором, а каждый класс объектов –именем класса. Каждый объект обладает определенным набором свойств. Для объектов одного класса набор этих свойств одинаков, а их значения, естественно, различаются.

Для изображения объектов и их свойств будем использовать следующие обозначения:

 - объект  - свойство

Связь между объектом и характеризующим его свойством будем изображать в виде линии. Связь может быть различной.

Объект может обладать только одним значением какого-то свойства (например, каждый человек может иметь только одну дату рождения). Такие свойства будем называть ***единичными***.

Для других свойств возможно существование одновременно нескольких значений у одного объекта (например, объект *Сотрудники* и его свойство *Иностранный язык*, человек может владеть несколькими иностранными языками). Такие свойства будем называть ***множественными***.

Для единичных свойств будем использовать одинарную стрелку ⟶, для множественных свойств двойную ⟶▸

Некоторые свойства являются постоянными, их значения не могут измениться с течением времени (например, *Дата рождения*); такие свойства будем называть ***статическими*** и обозначать буквой S над соответствующей линией.

Свойства, значения которых могут изменяться со временем (например, *Фамилия*, *Адрес*, *Телефон*), будем называть ***динамическими*** и обозначать буквой D.

Свойство может отсутствовать у некоторых объектов одного класса (например, свойство *Ученая степень*, не все объекты класса *Сотрудники* могут обладать указанным свойством). Такие свойства будем называть ***условными*** и изображать пунктирной

линией.

Существует понятие ***составного*** свойства (примеры таких свойств: *Адрес*, состоящий из «улицы», «дома», «квартиры»; *Дата рождения*, состоящая из «числа», «месяца», «года»). Для его обозначения будем использовать квадрат. В качестве примера на рис. 5.3 приводится изображение класса объектов *Сотрудники* и его свойств.

Рис. 5.3. Класс объектов и его свойства

В ИЛМ отображаются не отдельные экземпляры объектов, а классы объектов. Когда в ИЛМ изображено обозначение

объекта, то ясно, что речь идет о классе объектов, обладающих описанными свойствами. Поэтому в ИЛМ в большинстве случаев не вводят в явном виде еще и обозначение для класса объектов. Явное изображение последнего необходимо только в том случае, если в предметной области для данного класса объектов фиксируются не только характеристики, относящиеся к отдельным объектам этого класса, но и какие-то интегральные характеристики, относящиеся ко всему классу в целом. Например, если для класса объектов *Сотрудники* фиксируется не только возраст каждого из сотрудников, но и средний возраст всех сотрудников, то в ИЛМ необходимо отразить не только объект *Сотрудник*, но и класс объектов *Сотрудники* (рис. 5.4).

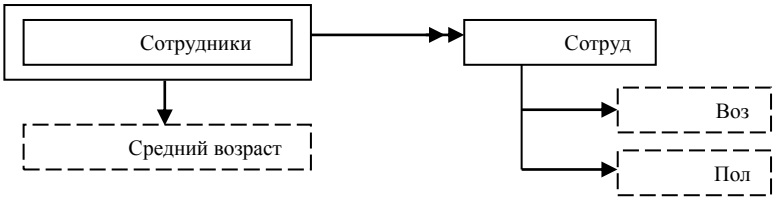


Рис. 5.4. Класса объектов
и интегральные характеристики класса

В инфологической модели фиксируются не только связи между объектом и его свойствами, но и связи между объектами разных классов.

Различают связи типа:

– один к одному (1:1);
– один ко многим (1:M);
– многие к одному (M:1);
– многие ко многим (M:M).

Иногда эти типы связей называют степенью связи.

Кроме степени связи, в ИЛМ для характеристики связи между разными объектами указывается **класс принадлежности**, который показывает, может ли отсутствовать связь объекта одного класса с каким-либо объектом другого класса. Различают обязательный и необязательный класс принадлежности. Объясним сказанное на конкретных примерах.

Пусть в ИЛМ отображается связь между двумя классами объектов: *Сотрудники* и *Язык иностранный*.

а) Предметной областью является завод, некоторые сотрудники которого знают иностранный язык, но ни один из них не владеет более чем одним языком.

Соответствующие диаграммы

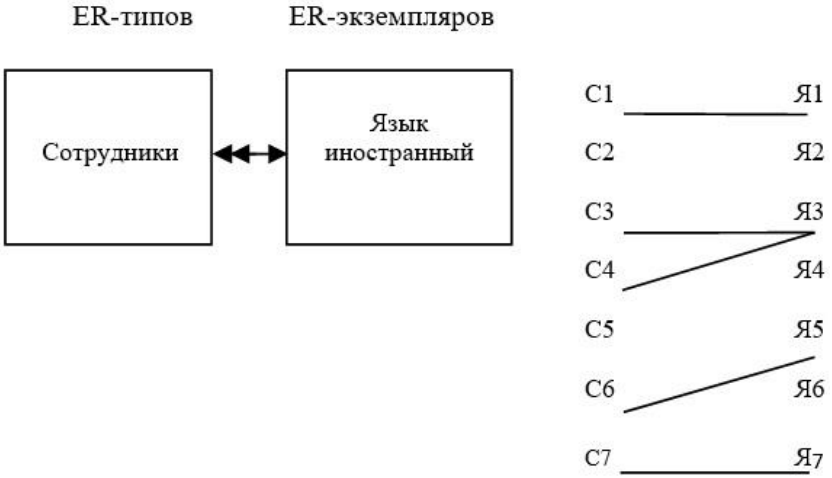


Рис. 5.5. Пример необязательного класса принадлежности

б) Предметной областью является институт, сотрудники которого обязательно должны владеть каким-либо иностранным языком, но никто не владеет более чем одним языком.

Соответствующие диаграммы

Рис. 5.6. Диаграмма обязательного класса принадлежности 1

В рассмотренных случаях между объектами наблюдается связь типа М:1; в случае *а*) класс принадлежности является необязательным для обоих объектов; в случае *б*) – для объекта *Сотрудники* класс принадлежности является обязательным, что изображается точкой в прямоугольнике.

в) Предметной областью является опять институт, некоторые сотрудники которого знают несколько иностранных языков.

Соответствующие диаграммы

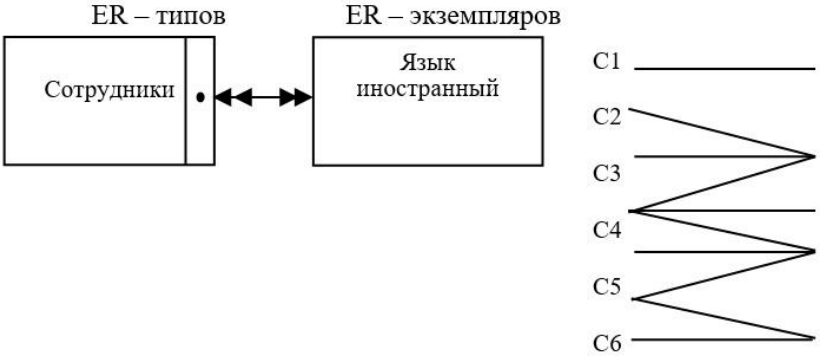


Рис. 5.7. Диаграмма обязательного класса принадлежности 2

В этом случае связь между объектами имеет тип M:M.

г) Предметной областью является лингвистический институт, каждый из сотрудников которого обязательно знает несколько иностранных языков, и по каждому из языков в этом институте имеется хотя бы один специалист, владеющий им. В этом случае связь между объектами будет M:M, и класс принадлежности обоих объектов является обязательным.

Примеры возможных ситуаций можно было бы продолжить, но суть ясна.

До этого момента мы рассматривали объекты, не вникая в их сложность. На самом деле объекты бывают простые и сложные.

Объект называют ***простым***, если он рассматривается как неделимый. ***Сложный*** объект представляет собой объединение других объектов, простых или сложных, также отображаемых в информационной системе.

Понятия простой и сложный являются относительными. При одном рассмотрении объект может считаться простым, а при другом этот же объект может рассматриваться как сложный. Например, объект *Стул* в подсистеме учета материальных ценностей будет рассматриваться как простой объект, а для предприятия, производящего стулья, это будет составной объект (включающий «ножки», «спинку», «сиденье»).

Сложные объекты подразделяют на составные, обобщенные и агрегированные.

Составной объект соответствует отображению связи «целое – часть». Примеры таких объектов: класс – ученики, группа – студенты и т.п.

Для отображения составных объектов в ИЛМ обычно не используются какие-либо специальные условные обозначения, а связь между составным и составляющими его объектами отображается так же, как это было описано выше (например, объекты *Группа* и *Студенты* связаны между собой отношением 1:М).

Обобщенный объект отражает наличие связи «род – вид» между объектами предметной области. Например, объекты *Студент*, *Школьник*, *Аспирант* образуют обобщенный объект *Учащиеся*. Объекты, составляющие обобщенный объект, называются его категориями. Как «родовой» объект, так и «видовые» объекты могут обладать определенным набором свойств. Причем «видовые» объекты обладают всеми теми свойствами, которыми обладает «родовой» объект, плюс свойствами, присущими только объектам этого вида.

Определение родовидовых связей означает классификацию объектов предметной области по тем или иным признакам. Подклассы могут выделяться в ИЛМ в явном виде (рис. 5.8).

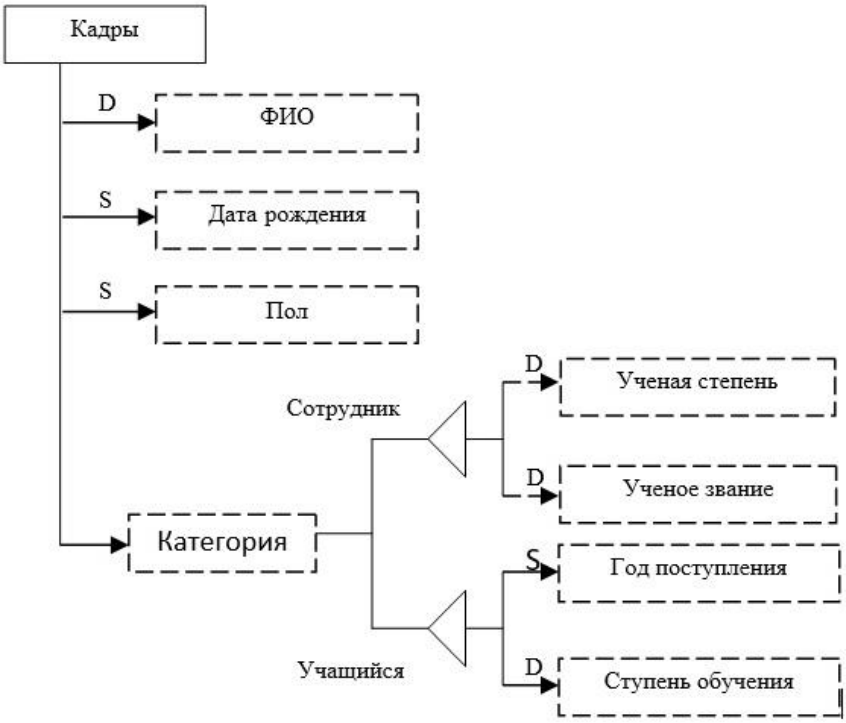


Рис. 5.8. Обобщенный объект

На рис. 5.8 изображен фрагмент ИЛМ, отражающий обобщенный объект *Кадры* для высшего учебного заведения. Для обобщенного объекта выделено две категории: *Сотрудник* и *Учащийся*. Для обозначения подкласса в схеме использован треугольник. Естественно, что классификация может быть многоуровневой. В рассматриваемом примере подкласс *Сотрудник*, в свою очередь, может быть классифицирован на *Преподаватель* и *Администрация*; *Учащийся* на *Студент* и *Аспирант*.

Агрегированный объект соответствует обычно какому-либо процессу, в который оказываются «вовлеченными» другие объекты. Например, агрегированный объект *Поставка* объединяет в себе объекты *Поставщик*, *Получатель*, *Продукт* и *Дата* (рис. 5.9). Для отображения агрегированного объекта в схеме использован ромб.

Рис. 5.9. Изображение агрегированного объекта

Агрегированный объект может так же, как и простой объект, иметь характеризующие его свойства. В рассматриваемом примере таким свойством является *Размер поставки.*

5.2. Даталогическое проектирование

Даталогическое проектирование является проектированием логической структуры БД, что означает определение всех информационных единиц и связей между ними, задание их имен и типов, а также некоторых количественных характеристик (например, длины поля).

При проектировании логической структуры БД осуществляется преобразование исходной инфологической модели в модель данных, поддерживаемую конкретной СУБД, и проверка адекватности полученной даталогической модели отображаемой предметной области.

При переходе от ИЛМ к ДЛМ следует иметь в виду, что первая включает в себя всю информацию о предметной области, необходимую и достаточную для проектирования БД. Это не означает, что все объекты, зафиксированные в ИЛМ, должны в явном виде отражаться в ДЛМ. Прежде чем строить ДЛМ, необходимо решить, какая информация будет храниться в базе данных.

При отображении объекта в таблицу идентификатор объекта будет являться полем этой таблицы, причем в большинстве случаев ключевым полем. В некоторых случаях

появляется необходимость введения искусственных идентификаторов, а именно:

1. Естественный идентификатор объекта не обладает свойством уникальности (например, среди сотрудников предприятия могут быть полные однофамильцы). В этом случае для однозначной идентификации объектов предметной области в информационной системе необходимо использовать искусственные коды.

2. Если объект участвует во многих связях, то для их отображения создается несколько таблиц, в каждой из которых повторяется идентификатор объекта. Чтобы не использовать во всех таблицах длинный естественный идентификатор объекта можно ввести и использовать более короткий искусственный код.

3. Если естественный идентификатор может изменяться со временем (например, фамилия), то это может вызвать много проблем, если, наряду с таким «динамическим» идентификатором, не использовать «статический» искусственный идентификатор.

6. НОРМАЛИЗАЦИЯ ОТНОШЕНИЙ

6.1. Сущность нормализации

Под ***нормализацией*** отношений понимают процесс построения оптимальной структуры таблиц и связей в реляционной базе данных. Теория нормализации основана на том, что определенный набор таблиц обладает лучшими свойствами при включении, модификации и удалении данных, чем все остальные наборы таблиц, с помощью которых могут быть представлены те же данные.

Нормализованным отношением называют отношение, каждый домен которого содержит только атомарные значения.

Отношение, даже если оно нормализовано, может обладать нежелательными свойствами – нормальная форма не обеспечивает сохранность набора отношений в процессе удаления, включения и обновления данных, ввиду существующей зависимости между последними, которая называется функциональной зависимостью (F-зависимостью).

Рассмотрим БД для консультанта университета, состоящую из одной таблицы *Консультант* со следующими полями: *Номер студента*, *ФИО студента*, *Номер комнаты*, *Номер телефона*, *Номер*

курса, *Семестр*[1], *Оценка*:

Таблица 6.1 БД *Консультант университета*

Номер студента	ФИО студента	Номер комнаты	Номер телефона	Номер курса	Семестр	Оценка
3215	Васильев О.И.	115	2136	ТВПС	5	4
3215	Васильев О.И.	115	2136	МСиС	5	4
3215	Васильев О.И.	115	2136	БД	6	5
3215	Васильев О.И.	115	2136	ТВПС	6	5
3462	Воловик В.А.	206	2344	ТВПС	7	5
3462	Воловик В.А.	206	2344	БД	6	5
3462	Воловик В.А.	206	2344	ООП	6	3
3567	Борисов И.Ю.	115	2136	ВМ	7	4
3567	Борисов И.Ю.	115	2136	ПиП	5	4
3567	Борисов И.Ю.	115	2136	ОС	5	3
4756	Гатаулин А.Е.	254	3321	ООП	4	5

Проблема включения. Когда у консультанта появляется новый консультируемый им студент, для него необходимо включить в БД кортеж с пустыми ячейками атрибутов: *Семестр*, *Оценка*, что повлечет за собой аномалии при поиске и редактировании данных (например, в результате запроса «Выдать список фамилий и номеров студентов, получивших хотя бы одну оценку ниже 3» в число таких студентов попадут такие, которые не закончили ни одного курса).

Проблема обновления. В отношении *Консультант* большое число избыточных данных. Избыточность (дублирование) данных всегда свидетельствует о возможности модификации только части требуемых данных с помощью опера операции обновления. Отношение характеризуется как явной, так и неявной избыточностью.

Явная избыточность: фамилия студента, его номер комнаты и номер телефона появляются в отношении не один раз. Например, если студент Васильев О.И. обратится к консультанту и сообщит ему об изменении номера своей комнаты, то консультант будет вынужден проследить изменение этого номера во всех четырех кортежах во избежание противоречивости данных.

[1] Семестр представляет собой семестр, в котором данный курс был завершен. Студент может повторить прохождение учебного курса и получить при этом другую оценку.

Неявная избыточность: один и тот же номер телефона имеют все студенты, живущие в одной комнате. Допустим, Васильев О.И. извещает консультанта о том, что его номер телефона изменен на 7777, забыв при этом сообщить о друге по комнате, консультант меняет телефонный номер в кортежах для Васильева О.И. – в результате правильный номер телефона друга будет фактически утерян.

6.2. Нормальные формы

Наличие тех или иных зависимостей в отношении определяет степень его нормализации.

Определение **первой нормальной формы** (1НФ): некоторое отношение r находится в 1НФ, если каждый его элемент имеет и всегда будет иметь атомарное значение. (Это определение просто устанавливает тот факт, что любое нормализованное отношение находится в 1НФ).

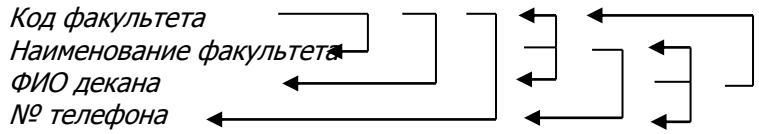
Примером отношения, находящегося в 1НФ, является отношение *Консультант*. На этом примере мы уже пояснили, почему отношение, находящееся в 1НФ, имеет нежелательную структуру.

Определение **второй нормальной формы** (2НФ): отношение r находится во 2НФ, если оно находится в 1НФ и, если каждый его атрибут, не являющийся основным атрибутом, функционально полно зависит от первичного ключа этого отношения.

Атрибут A_i отношения r называют **основным** атрибутом, если он является элементом первичного ключа данного отношения.

Рассмотрим отношение *Факультет* (*Код факультета*, *Наименование факультета*, *ФИО декана*, *Номер телефона*).

Функционально зависимы следующие атрибуты (исходим из предположения, что на одном телефонном номере может «висеть» несколько абонентов):

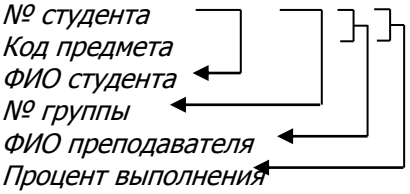


Таким образом, в рассматриваемом отношении три возможных ключа: *Код факультета*, *Наименование факультета*,

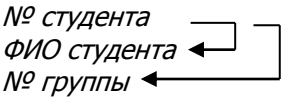
ФИО декана. Выберем в качестве первичного ключа атрибут *Код факультета*. Атрибуты *Наименование факультета*, *ФИО декана*, *№ телефона*, не являющиеся основными, функционально полно зависят от первичного ключа отношения, значит отношение находится во 2НФ.

Рассмотрим отношение *СтудентКурспроект (№ студента*, *Код предмета*, *ФИО студента*, *№ группы*, *ФИО преподавателя*, *Процент выполнения*). Предположим, что в одной группе могут учиться полные однофамильцы. Тогда у этого отношения один возможный ключ: *№ студента*, *Код предмета*. Атрибуты *ФИО студента* и *№ группы* не являются основными, но функционально зависят от основного атрибута *№ студента*, являющегося элементом ключа (т. е. отношение не находится во 2НФ).

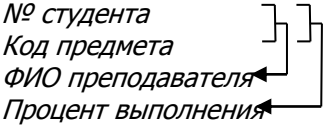
ФЗ между атрибутами данного отношения:



Разбив исходное отношение на два по принципу: атрибуты, функционально зависящие от одного основного атрибута, вместе с ним помещаются в одно отношение:



остальные атрибуты, которые полностью зависят от составного ключа, помещаются в другое отношение:



получим два отношения во 2НФ.

Отношения *Студент* (*№ студента*, *ФИО студента*, *№ группы*) и *Курспроект* (*№ студента*, *Код предмета*, *ФИО преподавателя*, *Процент выполнения*) являются проекциями исходного отношения, естественное соединение которых даст его. Таким образом, в процессе преобразования никакая информация не теряется, любая информация, которая может быть получена из первоначальной структуры, может быть получена и из новой структуры. Обратное, однако, неверно: новая структура может содержать информацию (например, о студентах, еще не получивших задания на курсовое проектирование), которая не может быть представлена в первоначальной структуре. В этом смысле новая структура является более точным отображением реального мира.

Приведение отношения ко 2НФ заключается в обеспечении полной функциональной зависимости всех неосновных атрибутов от первичного ключа за счет разбиения отношения на несколько.

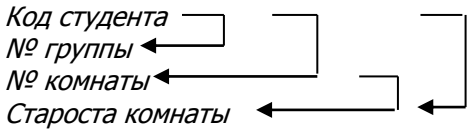
Определение **третьей нормальной формы** (3НФ): отношение r находится в 3НФ, если оно является отношением во 2НФ, и каждый его атрибут, не являющийся основным, не транзитивно зависит от первичного ключа этого отношения.

Транзитивная зависимость определяется следующим образом: если $X \rightarrow Y$ и $Y \rightarrow Z$, то $X \rightarrow Z$ (Z транзитивно зависит от X).

Приведение отношения к 3НФ заключается в устранении транзитивных зависимостей в отношении путем разбиения исходного отношения r(X, Y, Z) на: r1(X, Y) и r2(Y, Z).

Рассмотрим отношение *Общежитие* (*Код студента*, *№ группы*, *№ комнаты*, *Староста комнаты*).

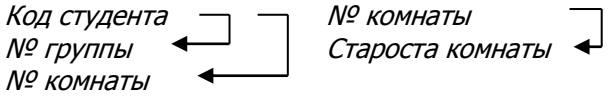
F-зависимости между атрибутами данного отношения:



Возможным и единственным ключом отношения является атрибут *Код студента*. Отношение находится во 2НФ, но не в третьей, так как неосновной атрибут *Староста комнаты* зависит от атрибута *№ комнаты*, который, в свою очередь, зависит от ключа отношения *Код студента*, и, следовательно, атрибут *Староста комнаты* транзитивно зависит от ключа. Данное отношение имеет недостатки (например, если студент переходит жить в другую

комнату, то работающему с данной БД необходимо, кроме номера комнаты, не забыть изменить и старосту комнаты).

Для устранения недостатков разобьем отношение на следующие два:



Полученные отношения находятся в 3НФ и они предпочтительнее исходного (например, информация о старосте комнаты может потребоваться независимо от информации о студентах, проживающих в этой комнате).

Отношение может быть в 3НФ и при этом все же иметь некоторые нежелательные свойства.

Например, рассмотрим отношение *ГАИ* (*город*, *адрес*, *индекс*), где *индекс* – это индекс отделения связи, *адрес* – название улицы, номер дома и номер квартиры.

F-зависимости между атрибутами отношения:

То есть полный адрес определяет почтовый индекс, а тот, в свою очередь, определяет название города, но не определяет адрес.

Ключом отношения является составной атрибут: *город*, *адрес*. Отношение находится в 3НФ (в нем нет транзитивных и неполных зависимостей).

В рассматриваемое отношение нельзя включить кортеж для города, к которому относится заданный индекс, если неизвестен адрес с этим индексом. Если же удалить все кортежи, содержащие адреса в одном городе, то будет потерян индекс данного города.

Для устранения недостатков 3НФ вводится четвертая нормальная форма, имеющая две разновидности.

Первая разновидность известна под названием **нормальной формы Бойса – Кодда** (НФБК) (в некоторых литературных источниках она называется усовершенствованной 3НФ).

Определение: отношение r находится в НФБК, если и

только если каждый детерминант отношения является возможным ключом.

Детерминантом отношения называют атрибут (возможно, составной) от которого функционально полно зависит другой атрибут.

Пример отношения в НФБК: *ВыдачаВозврат* (*Код книги*, *Код читателя*, *Дата выдачи*, *Дата возврата*).

Функциональные зависимости, имеющие место в данном отношении:

Функциональные зависимости определены из следующих соображений: любая книга берется (соответственно возвращается) не один раз и разными читателями (т.е. по любому атрибуту данного отношения нельзя однозначно определить никакой другой атрибут); любой читатель может взять одну книгу дважды (т.е. по коду книги и по коду читателя нельзя однозначно определить дату выдачи книги); в один день читатель может взять (соответственно отдать) несколько книг (т.е. по коду читателя и дате выдачи (дате возврата) книги нельзя однозначно определить код книги). Таким образом, имеем два детерминанта, составные атрибуты: *Код книги*, *Дата выдачи* и *Код книги*, *Дата возврата* и они же являются возможными ключами отношения.

Может оказаться так, что отношение в 3НФ нельзя будет привести к НФБК без потери зависимостей между атрибутами этого отношения.

Обратимся, например, к отношению *ГАИ* (*город*, *адрес*, *индекс*), которое не находится в НФБК, так как имеет место зависимость *индекс* → *город*.

Это отношение можно разбить на отношения: *ГА* (*город*, *адрес*) и *АИ* (*адрес*, индекс), но тогда зависимость *индекс* → *город* будет утеряна.

Определение второй разновидности **четвертой нормальной формы** (4НФ): отношение r находится в 4НФ тогда и только тогда, когда при существовании многозначной зависимости в r атрибута Y от атрибута X, все остальные атрибуты r функционально зависят от X.

Атрибут X **многозначно** определяет атрибут Y, если с

каждым значением *x* может использоваться значение *y* из фиксированного подмножества значений Y. Обозначается: X ↠ Y.

Существуют НФ высшего порядка. В качестве примера рассмотрим отношение *КПУ* (*Курс*, *Преподаватель*, *Учебник*), где *Курс* – наименование курса, *Преподаватель* – фамилия преподавателя, *Учебник* – название учебника. Пусть данному курсу может соответствовать любое число преподавателей и любое количество учебников, и пусть преподаватели и учебники не зависят друг от друга.

Экземпляр отношения *КПУ*, касающийся только одного курса:

Курс	Преподаватель	Учебник
Физика	Иванов	Основы механики
Физика	Иванов	Законы оптики
Физика	Петров	Основы механики
Физика	Петров	Законы оптики

Атрибут *Курс* многозначно определяет атрибут *Преподаватель* и имеется многозначная зависимость атрибута *Учебник* от атрибута *Курс*:

Курс ↠ *Преподаватель*;
Курс ↠ *Учебник*.

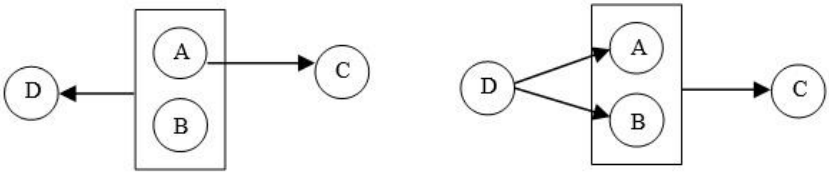
Рассматриваемое отношение находится в 3НФ (оно все является ключом). Недостатком данного отношения является, например, то, что при добавлении информации о том, что для курса физики используется новый учебник «Современная механика», необходимо будет создать несколько кортежей, по одному для каждого из преподавателей.

Для устранения недостатков отношения, представленного в 3НФ, выполняют разложение по многозначным зависимостям данного отношения, т. е. отношение r (A, B, C), где A ↠ B и A ↠ C, разбивают на два: r1 (A, B) и r2 (A, C).

Для нашего примера: *КП* (*Курс*, *Преподаватель*), *КУ* (*Курс*, *Учебник*). Каждое из полученных отношений находится в четвертой нормальной форме.

Приведем примеры выявления нормальной формы отношений.

Дана диаграмма функциональных зависимостей для некоторого отношения (варианты *а*, *б*):



а) *б*)

Необходимо определить, в какой нормальной форме находится данное отношение.

Для варианта *а*): имеем две функциональные зависимости: A → C и A, B → D, т.е. ключом отношения является составной атрибут A, B. Будем считать, что каждый из имеющихся атрибутов имеет атомарное значение. Тогда рассматриваемое отношение находится в 1НФ. Отношение не находится во 2НФ, в силу имеющейся функциональной зависимости неосновного атрибута C от атрибута A – части ключа.

Для варианта *б*: имеем три функциональные зависимости: D → A, D → B и A, B → C, т.е. ключом отношения является атрибут D. Будем считать, что каждый из имеющихся атрибутов имеет атомарное значение. Тогда рассматриваемое отношение находится в 1НФ. Отношение находится во 2НФ, так как все неосновные атрибуты (A, B и C) функционально полно зависят от ключа отношения. Отношение не находится в 3НФ, так как неосновной атрибут C транзитивно зависит от ключа D: D → A, B → C.

7. БЕЗОПАСНОСТЬ БАЗ ДАННЫХ И ОБЕСПЕЧЕНИЕ ИХ ЦЕЛОСТНОСТИ

Актуальными являются вопросы безопасности и целостности БД. Защита от несанкционированного доступа, изменения или разрушения данных, осуществляется созданием системы безопасности и корректное выполнение операции с БД. Различные пользователи должны обладать различными полномочиями доступа к БД. Защита БД, т.е. обеспечение

защищенности базы данных против любых предумышленных или непредумышленных угроз, осуществляется с помощью различных компьютерных и некомпьютерных средств.

В *Access* реализованы следующие способы защиты: парольная защита, защита на уровне пользователя, шифрование.

Целостность данных обеспечивается набором специальных предложений, называемых ограничениями целостности.

Ограничения целостности представляют собой утверждения о допустимых значениях отдельных информационных единиц и связях между ними.

В большинстве случаев ограничения целостности определяются особенностями предметной области (хотя могут отражать и чисто информационные характеристики).

Ограничения целостности могут относиться к разным информационным объектам: атрибутам, кортежам, отношениям, связям между ними и т.д.

Для полей (атрибутов) используются следующие виды ограничений:

– Тип и формат поля (автоматически допускают ввод только данных определенного типа).

– Задание диапазона значений. Данное ограничение используется обычно для числовых полей. Различают открытые и закрытые диапазоны: первые фиксируют значение только одной из границ, вторые – обеих границ.

– Недопустимость пустого поля. Ограничение позволяет избежать появления в БД «ничейных» записей, в которых пропущены какие-либо обязательные данные, например: поле «ФИО» должно обязательно иметь значение, а у поля «Ученая степень» значение может отсутствовать.

– Задание домена. Ограничение позволяет избежать излишнего разнообразия данных, если его можно ограничить, например, значением поля «должность» для преподавателей может быть одно из следующих значений: ассистент, старший преподаватель, доцент, профессор.

– Проверка на уникальность значения какого-либо поля. Ограничение позволяет избежать записей-дубликатов. Данное ограничение возникает при отображении в базе данных каких-либо объектов, когда уникальное поле является идентификатором объекта; поэтому данное ограничение часто называют **ограничением целостности объекта**.

Приведенная классификация видов ограничений является довольно условной, поэтому задание диапазона также можно

считать способом задания домена.

Перечисленные выше ограничения определяют проверку значения поля вне зависимости от того, вводится ли это ограничение впервые или корректируется имеющееся в БД значение. Ограничения, используемые только при проверке допустимости корректировки, называют **ограничениями перехода**. Например, если в БД имеется поле «Возраст сотрудника», то при корректировке значение этого поля может только увеличиваться; если в БД имеется поле «Год рождения», то на корректировку этого поля должен быть наложен запрет. В случае попытки произвести некорректное исправление должно выдаваться диагностирующее сообщение.

Что касается ограничения целостности, относящегося к кортежам, то здесь имеется в виду либо ограничение на значение всей строки, рассматриваемой как единое целое (естественным ограничением является требование уникальности каждой строки таблицы), либо ограничения на соотношения значений отдельных полей в пределах одной строки (например, значение поля «Стаж» не должно превышать «Возраст»).

Существуют ограничения, проверяющие соотношения между записями одной таблицы, например, «Год рождения матери» должен быть меньше, чем «Год рождения ребенка»; нельзя быть родителем и ребенком одного и того же человека.

В качестве примера ограничений, относящихся ко всей таблице, можно привести следующий. Предположим, что фонд заработной платы формируется исходя из величины средней заработной платы одного сотрудника, которая составляет 10 000 р. Тогда в качестве ограничения целостности таблицы может быть задано выражение, указывающее, что среднее значение поля «Оклад» должно быть не больше 10 000.

Если СУБД не позволяет контролировать какие-либо ограничения целостности, то следует создавать процедуры (программы), позволяющие это делать.

Все ограничения, которые были рассмотрены выше, затрагивают информационные единицы в пределах одной таблицы. Кроме такого рода ограничений имеются ограничения, относящиеся к нескольким взаимосвязанным таблицам, например, **ограничение целостности связи**, которое выражается в том, что значение атрибута, отражающего связь между объектами и являющегося внешним ключом отношения, обязательно должно совпадать с одним из значений атрибута, являющегося ключом отношения, описывающего соответствующий объект. Например, в

БД имеются три таблицы: «Преподаватели», «Дисциплины» и таблица, отражающая связь между преподавателями и дисциплинами: код преподавателя в последней из трех таблиц должен соответствовать одному их кодов в таблице «Преподаватели», а код дисциплины – значению соответствующего поля в таблице «Дисциплины».

Своеобразным видом ограничения является **запрет на обновление**. Он может относиться и к отдельному полю и ко всей записи, и к целой таблице. Например, не могут меняться значения таких полей, как «Дата рождения», «Место рождения»; или пусть имеется таблица «Поощрения» с полями: «Табельный номер сотрудника», «Вид поощрения», «Дата», – в такую таблицу записи могут только добавляться, а изменяться не могут.

В рассматриваемом примере наблюдается также ограничение по существованию между таблицей «Поощрения» и таблицей «Сотрудники»: табельный номер в таблице «Поощрения» должен обязательно присутствовать в таблице «Сотрудники»; при удалении записи в таблице «Сотрудники» все связанные с ней записи в таблице «Поощрения» также должны быть удалены.

Ограничения целостности разделяют по моменту контроля за соблюдением ограничения на одномоментные и отложенные. **Отложенные ограничения** целостности могут не соблюдаться в процессе выполнения какой-либо группы операций, но обязаны быть соблюдены по завершении выполнения этой группы операций. С понятием отложенного ограничения тесно связано понятие транзакции. В системе *Microsoft Access* транзакция определяется как набор операций, результат которых подтверждается (сохраняется) в том и только в том случае, если все операции набора прошли успешно. Если какая-либо из операций транзакции не выполнена, то все выполненные ранее операции отменяются, и данные возвращаются к тому состоянию, которое они имели до начала выполнения транзакции. Примером может служить перевод денег с одного банковского счета на другой, состоящий из двух операций: удаление денег с одного счета и добавление такой же суммы денег на другой счет.

Ограничения целостности разделяют по способу задания – на явные и неявные. **Неявные ограничения** определяются спецификой модели данных и проверяются СУБД автоматически. Неявные ограничения обычно относятся к классу синтаксических ограничений в отличие от семантических ограничений целостности, обусловленных спецификой предметной области.

Рассмотренные примеры ограничений целостности относятся к данным пользователя. Понятие же целостности может относиться и к служебной информации.

Наряду с понятием целостности БД, может быть введено понятие информационной целостности банка данных (БнД), заключающееся в обеспечении правильности взаимосвязи всех его информационных компонентов.

Резюме. Задание ограничений целостности и их проверка являются важной частью проектирования и функционирования БнД. При проектировании БнД необходимо изучить, какие возможности по контролю целостности предоставляет используемая СУБД. Если СУБД автоматически не поддерживает нужное ограничение, то обеспечение его соблюдения становится заботой проектировщика.

8. ИСПОЛЬЗОВАНИЕ БАЗЫ ДАННЫХ И МЕТОДА ЭКСПЕРТОВ НА ТРАНСПОРТЕ

Рассмотрим деятельность виртуального «Мультибрендового автосалона» в рамках компьютерной модели с использованием возможностей *MS ACCESS*. Выделяем существенные критерии конкретной ситуации, когда человек, желающий приобрести автомобиль в салоне или на вторичном рынке интересуется характеристиками автомобиля, мнением специалистов, знакомых, которые эксплуатировали различные марки автомобилей. Правильный выбор критериев играет существенную роль в принятии оптимального решения. В теории принятия решений не найдено общего метода выбора критериев оптимальности. В основном руководствуются опытом или рекомендациями.

Каждый студент создает БД для выбранной марки автомобиля. Марки автомобилей не должны повторяться.

Клиенту салона предоставляется возможность познакомиться с параметрами автомобиля любой марки в заданной ценовой категории, просмотреть фото моделей и изучить мнения экспертов (рис. 8.1, 8.2). Для определения наиболее важных факторов и принятия обоснованных решений необходимо опираться на опыт, знания и интуицию специалистов. Студентам рекомендовано применить метод экспертов.

После Второй мировой войны в рамках кибернетики, теории управления, менеджмента и исследования операций стала

развиваться самостоятельная дисциплина – теория и практика экспертных оценок. В данном случае эксперты – такие люди, которые обладают знаниями по устройству и свойствам различных марок автомобилей, правилам дорожного движения, навыками вождения.

Все объекты Access

Поиск...

Таблицы

- Mazda 3
- Mazda 6
- Mazda CX-5
- Mazda CX-9
- Общее

Формы

- Общее

Общее | Mazda CX-9 | Mazda 3

Марки авто	Тип кузова	Модельный год	Длина, мм	Ширина, мм	Количество дверей	Объем багажника, л	Тип двигателя
Mazda 323 FA AT 1,3	Универсал	1977	4000	1595	5	320	Бензиновый
Mazda 323 FA MT 1,0	Хетчбек	1977	3820	1595	3	300	Бензиновый
Mazda 323 FA MT 1,3	Хетчбек	1977	3820	1595	5	300	Бензиновый
Mazda 323 BD MT 1,1	Хетчбек	1980	3960	1630	3	330	Бензиновый
Mazda 323 BD MT 1,3	Хетчбек	1980	3960	1630	5	330	Бензиновый
Mazda 323 BD MT 1,5	Хетчбек	1980	3960	1630	5	330	Бензиновый
Mazda 323 BA MT 1,5	Хетчбек	1994	4245	1710	5	346	Бензиновый
Mazda 323 BA AT 1,5	Седан	1994	4245	1710	3	346	Бензиновый
Mazda 323 BA MT 1,8	Седан	1994	4245	1710	3	346	Бензиновый
Mazda 3 Sedan I MT 1,6	Седан	2003	4490	1755	5	413	Бензиновый
Mazda 3 Sedan I 1.6 AT	Седан	2003	4490	1755	5	413	Бензиновый
Mazda 3 Sedan I 2.0 MT	Седан	2003	4490	1755	5	413	Бензиновый
Mazda 3 II 1.6 MT	Хетчбек	2011	4460	1755	5	340	Дизельный
Mazda 3 II 1.6 AT	Хетчбек	2011	4460	1755	5	1360	Бензиновый
Mazda 3 II 2.0 MT	Хетчбек	2011	4460	1755	5	1360	Бензиновый

Общее | Mazda CX-9 | Mazda 3

Объем двигателя, л	Цена, руб	Скорость, км/ч	Разгон до 100 км/ч	Расход топлива на 100 км, л	Класс автомобиля	КПП
1,3	90 000,00р.	140	-	8	C	АКПП
1,0	50 000,00р.	140	-	6,7	C	МКПП
1,3	70 000,00р.	150	-	6,8	C	МКПП
1,1	80 000,00р.	145	14,8	6,5	B	МКПП
1,3	90 000,00р.	153	12	6,7	B	МКПП
1,5	95 000,00р.	160	11,6	7,3	B	МКПП
1,5	98 000,00р.	175	10,9	7,8	C	МКПП
1,5	98 000,00р.	161	14,2	8,4	C	АКПП
1,8	100 000,00р.	191	9,5	8,0	C	МКПП
1,6	200 000,00р.	185	11,5	6,9	C	МКПП
1,6	230 000,00р.	177	12,5	7,6	C	АКПП
2,0	300 000,00р.	209	9,1	7,9	C	МКПП
1,6	600 000,00р.	181	12,1	6,5	C	МКПП
1,6	650 000,00р.	173	13,1	7,5	C	АКПП
2,0	700 000,00р.	203	9,1	7,4	C	МКПП

Рис. 8.1. База данных автомобиля Mazda

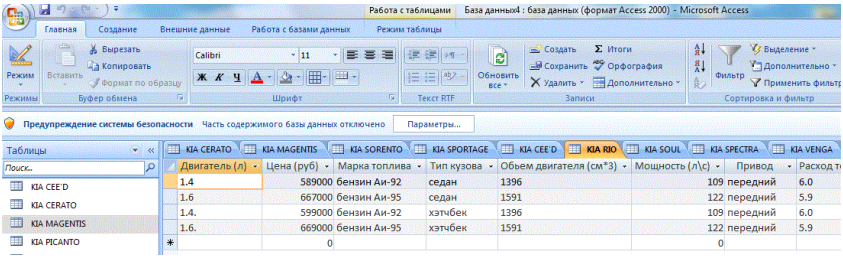


Рис. 8.2. База данных автомобиля KIA

Для достижения требуемой вероятности правильного решения выбираются 10–15 экспертов: это преподаватели кафедры СТЭАС, сотрудники автошколы, автосалонов, сервисных центров. Экспертный метод предполагает использование всей предварительной информации об изучаемой марке автомобиля и ее моделях.

Процедура проведения экспертного метода состоит из 4-х этапов:

– опрос и анкетирование по рассматриваемой модели;

– статистическая обработка анкет и составление матрицы рангов;

– оценка согласованности мнений специалистов;

– анализ диаграммы рангов и исключение незначащих факторов.

Задача каждой группы:

– выявление основных факторов, влияющих на желание человека приобрести ту или иную марку автомобиля в заданной ценовой категории;

– сбор субъективных оценок факторов каждым экспертом

группы, определение коэффициента весомости факторов;

– определение нормированных коэффициентов весомости факторов;

– определение средней значимости факторов.

Статистические выводы могут быть адекватны реальности только тогда, когда они не зависят от того, какую единицу измерения предпочтет исследователь, т.е. когда они инвариантны относительно допустимого преобразования шкалы. Для сравнения различных критериев проще всего воспользоваться средними значениями. Так как известны различные виды средних величин (среднее арифметическое, медиана, мода, среднее геометрическое, среднее гармоническое, среднее квадратичное), то выбор метода обработки остается за студентом [3]. Обобщением нескольких из перечисленных методов является среднее по Колмогорову. Для чисел X1, X2,...,Xn среднее по Колмогорову вычисляется по формуле

$$G\{(F(X1)+F(X2)+...F(Xn))/n\}, \quad (8.1)$$

где F – строго монотонная функция; G – функция, обратная к F.

Обычно эксперту предлагается оценить 5–12 различных факторов. При большом числе факторов снижается объективность оценок. По выявленным факторам каждый эксперт проставляет оценки по десяти бальной шкале индивидуально, занося их в табл. 8.1.

Таблица 8.1 Коэффициент весомости факторов

Номер фактора	Эксперты											
	1	2	3	4	5	6	7	8	9	10	11	12
1	1	5	2	5	5	0	0	1	1	3	3	2
2	3	2	2	3	2	1	3	2	4	5	2	3
3	10	10	8	5	5	6	10	10	10	10	7	9
4	2	4	1	1	1	1	1	2	2	3	1	1
5	4	5	6	7	2	5	7	1	2	4	7	5
6	5	7	6	5	9	3	3	4	4	5	4	7
7	4	5	8	9	10	3	3	3	8	8	7	1
8	4	3	1	4	8	7	4	4	6	7	7	5
9	6	5	6	3	6	7	8	9	9	5	8	6
10	10	8	4	10	4	9	6	4	7	5	5	2
Сумма	49	54	45	52	52	42	45	40	53	55	51	41

Суммарные оценки в табл. 8.1 изменяются от 40 до 55. Это связано с тем, что, соблюдая пропорциональное соотношение при определении значимости факторов, одни эксперты сравнивали их по минимальной – личной шкале, а другие по максимальной шкале. На следующем этапе работы группы происходит определение коэффициентов конкордации для каждого эксперта и всей экспертной группы в целом с построением диаграмм удельных весов факторов, оценкой согласованности мнений экспертов и исключением незначащих факторов. Для сравнительной оценки показаний экспертов удельные веса всех факторов нормируются в соответствии с формулой (8.2) и заносятся в табл. 8.2:

$$a_{ij} = \frac{a'_{ij}}{\sum_{i=1}^{n} a'_{ij}}, \qquad (8.2)$$

где a'_{ij} – коэффициент весомости i-го фактора, назначенный j-тым экспертом;

$\sum_{i=1}^{n} a'_{ij}$ – сумма всех коэффициентов весомости для n факторов, назначенная j-тым экспертом.

Таблица 8.2 Нормированные коэффициенты весомости факторов

Номер фактора	Эксперты												Средняя значимость фактора
	1	2	3	4	5	6	7	8	9	10	11	12	
1	0,03	0,083	0,044	0,086	0,086	0	0	0,035	0,028	0,055	0,587	0,048	0,090
2	0,06	0,037	0,044	0,058	0,038	0,034	0,067	0,05	0,075	0,082	0,038	0,073	0,055
3	0,3	0,285	0,278	0,086	0,086	0,243	0,333	0,35	0,238	0,233	0,237	0,33	0,250
4	0,04	0,074	0,033	0,028	0,028	0,034	0,033	0,05	0,038	0,055	0,03	0,034	0,040
5	0,08	0,083	0,233	0,235	0,038	0,228	0,256	0,035	0,038	0,073	0,237	0,233	0,147
6	0,2	0,23	0,233	0,086	0,273	0,072	0,067	0,2	0,075	0,082	0,078	0,272	0,156
7	0,08	0,083	0,278	0,273	0,283	0,072	0,067	0,075	0,252	0,245	0,237	0,034	0,165
8	0,08	0,056	0,044	0,077	0,26	0,267	0,088	0,2	0,223	0,237	0,237	0,233	0,167
9	0,23	0,083	0,233	0,058	0,225	0,267	0,277	0,335	0,27	0,082	0,257	0,246	0,214
10	0,3	0,248	0,088	0,283	0,077	0,324	0,233	0,2	0,233	0,082	0,088	0,048	0,184

Для оценки согласованности мнений экспертов вычисляется коэффициент конкордации как алгебраическая разность между значениями коэффициентов весомости, установленных j-тым экспертом, и их средней величиной по каждому из факторов:

$$K_j = \frac{1}{2}\sum_{i=1}^{n} | a_{ij} - \bar{a}_i |, \qquad (8.3)$$

где $\bar{a}_i$ – средняя значимость i-го фактора.

Среднее отклонение по всей экспертной группе будет равно

$$K = \frac{\sum_{j=1}^{N} K_j}{N} = 0{,}1472,$$

где N – число экспертов.

Оценка согласованности мнений экспертов характеризуется величиной D=1-K. При D=1 принимается полная согласованность мнений. В нашем случае D=0,8388 соответствует практически полной согласованности мнений. Анализ диаграммы нормированных удельных весов факторов показывает, что степени влияния факторов упорядочены и существенно отличны (рис. 8.3).

Рис. 8.3. Диаграмма удельных весов факторов, влияющих на исследуемый процесс

При проведении дальнейших исследований можно не учитывать влияние тех факторов, которые получили наименьшую значимость.

Приведем примеры мнений специалистов для автомобилей и сравнительную характеристику автомобилей HYUNDAI (табл. 8.3 – 8.5).

Таблица 8.3 Мнение экспертов 1

	A	B	C	D	E	F	G
1	Наименование	Цена	Двигатель	Тип топлива	Максимальная скорость	Емкость бака	Тип привода
2	ASX	6	8	8	8	7	8
3	I-MIEV	6	5	5	6	5	7
4	L200	5	8	8	7	7	9
5	LANCER	6	5	8	7	7	5
6	OUNLANDER	5	8	6	8	7	9
7	PAJERO	4	8	6	7	8	9

Таблица 8.4 Мнение экспертов 2

Модель	Дешевизна обслуживания /содержания		Надёжность		Управляемость		Динамика и расход топлива	
	кузов	ходовая ДВС,КПП и т.д.	кузов	ходовая ДВС,КПП и т.д.	обычная (1-5)	хорошая (5-10)	динамика	расход
Elantra	4	5	7	6	4		5	7
Equus	4	5	4	3	3		7	5
Genesis	3	7	3	4		5	6	5
Grandeor	6	5	6	6		5	5	4
H-1	8	9	8	8		7	8	2
Ix35	5	6	4	7	4		7	3
Ix55	5	4	6	4	4		5	3
Santa Fe	3	4	5	4		6	6	3
Sonata	3	5	4	5		6	4	5
Veloster	7	6	5	5		6	4	4

Таблица 8.5 Сравнительная характеристика Hyundai

Параметры	Марка автомобиля		
	Hyundai Sonata	Hyundai Elantra	Hyundai Ix35
Двигатель	У 2,4-литровых двигателей возможен выход из строя датчик положения коленвала	Конструктивная особенность – свист от ремней (не «лечится»). Ненадёжный радиатор охлаждения (замена – 9000 руб.)	Ресурс ремня ГРМ бензинового двигателя 2.7 – 60 тыс. км (комплект ремня ГРМ – 3 880 руб.; работа – 1,7 нормо-часа)
Трансмиссия	минимальные проблемы с низкой стоимостью ремонта	минимальные проблемы с низкой стоимостью ремонта	минимальные проблемы с низкой стоимостью ремонта
Система управления и подвеска	При частой загрузке багажника быстро выходят из строя стойки стабилизаторов (стойка стабилизатора задней подвески – 161 руб.; работа – 0,5 нормо-часов) и сайлентблоки рычагов (сайлентблок рычага – 202 руб.; работа – 1,2 нормо-часа)	Ресурс оригинальных амортизаторов – 50 тыс. км.	Ресурс сайлентблоков подрамника составляет 60 тыс. км. (сайлентблок подрамника – 596 руб.; работа – 2,5 нормо-часа). Ресурс подшипников трансмиссии – 50 тыс. км (подшипник трансмиссии – 2900 руб.; работа – 1,7 нормо-часов). После 100 тыс. км потребуется замена гофры выпуска (гофра выпуска – 575 руб.; работа – 1,5 нормо-часа)
Тормозная система	минимальные проблемы с низкой стоимостью ремонта	Ресурс оригинальных тормозных дисков – около 30 тыс. км	минимальные проблемы с низкой стоимостью ремонта
Подогрев воздух и кондиционер	минимальные проблемы с низкой стоимостью ремонта	минимальные проблемы с низкой стоимостью ремонта	минимальные проблемы с низкой стоимостью ремонта
Система запуска и зарядки	минимальные проблемы с низкой стоимостью ремонта	Ненадёжные оригинальные стартеры – лучше заменить на неоригинальный	минимальные проблемы с низкой стоимостью ремонта

Электрические компоненты и прочее	минимальные проблемы с низкой стоимостью ремонта	минимальные проблемы с низкой стоимостью ремонта	минимальные проблемы с низкой стоимостью ремонта
Устойчивость кузова к коррозии		Возможна коррозия на нижней кромке дверей	минимальные проблемы с низкой стоимостью ремонта

9. ПРАКТИЧЕСКИЕ ЗАДАНИЯ И ТЕСТЫ

Задание «Реализация реляционной базы данных в СУБД Access»:

Создать таблицы, определив для каждого поля таблицы свойства. Для полей внешнего ключа создать поле с подстановкой (раскрывающий список).

Установить связи между таблицами.

Создать запросы:

– три запроса на выборку со сложными критериями отбора;

– три запроса, использующие групповые операции и статистические функции SQL;

– параметрический запрос;

– перекрестный запрос;

– запросы на изменение.

Создать две формы: простую (на основе одной таблицы) и сложноподчиненную (на основе двух таблиц, объединенных связью «один-ко-многим»). Разместить в формах различные элементы управления: поля с раскрывающимися списками, кнопки для запуска запросов и форм и т. д.

9.1. Решение задач с использованием баз данных

Для успешного решения задач по теме «Технология хранения, поиска и сортировки информации в базах данных» необходимы прочные знания принципов организации реляционных баз данных, структуры и элементов таблицы. Важно отличать понятия «запись» и «поле», знать определение первичного ключа таблицы, владеть методами сортировки и выборки данных, а также обязательно знание логических операций.

Задача 1

В олимпиаде по информатике принимали участие 10 студентов 1-го курса. Известно, что задания в игре делились на три типа (A, B, C). Задания части A оценивались 3 баллами, части B – 4 баллами, части C – 5 баллами. Победителем объявляется тот, кто набрал больше всего баллов.

Какой запрос необходимо составить, чтобы определить победителя, если для каждого студента известно количество правильно выполненных заданий из каждой части игры.

Таблица 9.1 БД олимпиады по информатике

Фамилия, Имя	часть А	часть В	часть С
Авакумова Елена	4	4	1
Величко Марина	10	2	0
Гайденко Михаил	9	6	1
Дьячков Артур	10	9	9
Митрофанова Анна	10	8	3
Никофоров Максим	10	9	7
Петренко Олег	10	10	4
Трофимова Алина	10	9	8
Семенов Леонид	9	5	7
Срубизин Роман	7	3	0

1. Отсортировать таблицу по возрастанию значений выражения 3*часть А+ 4* часть В+ 5* часть С, а затем взять первую строку.
2. Отсортировать таблицу по убыванию значений выражения 3*часть А+ 4* часть В+ 5* часть С, а затем взять первую строку.
3. Отсортировать таблицу по возрастанию значений выражения часть А+ часть В+ часть С, а затем взять последнюю строку.
4. Отсортировать таблицу по возрастанию значений выражения часть А+ часть В+ часть С, а затем взять первую строку.

Решение

Очевидно, что для определения суммарного количества баллов каждого участника игры нужно использовать выражение 3*часть А+ 4* часть В+ 5* часть С, затем отсортировать по убыванию и взять первую строку. Значит, верен второй вариант.

Ответ: 2.

Задача 2

В табличной форме представлен фрагмент базы данных о результатах тестирования учащихся в 100-балльной системе:

Таблица 9.2 Результаты тестирования

Фамилия	Пол	Математика	Русский язык	Химия	Информатика	Биология
Агапова	ж	82	56	46	32	70
Воронин	м	43	62	45	74	23
Григорчук	м	54	74	68	75	83
Роднина	ж	71	63	56	82	79
Сергеенко	ж	33	25	74	38	46
Черепанова	ж	18	92	83	28	61

Сколько записей в данном фрагменте удовлетворяют условию

Пол = 'м' ИЛИ Химия>Биология:

1) 5 2) 2 3) 3 4) 4

Решение

1. Условие «Пол = 'м' или Химия>Биология» означает, что выборке подлежат записи, в которых поле «Пол» содержит значение 'м' или значение поля «Химия» больше значения поля «Биология».

2. Рассмотрим первую запись базы данных, содержащую сведения об Агаповой.

У Агаповой пол женский, балл по химии меньше балла по биологии (46<70). Получаем дизъюнкцию двух ложных высказываний. Результат – ложь.

3. Рассмотрим вторую запись базы данных, содержащую сведения о Воронине. В этом случае получается дизъюнкция двух истинных высказываний. Результат – истина.

4. Третья запись содержит сведения о Григорчуке. Первое условие выполняется (пол мужской), а второе нет (68 < 83). Дизъюнкция истинного и ложного высказывания – истина.

5. В четвертой записи получается дизъюнкция двух ложных высказываний, результат – ложь.

6. В пятой и шестой записях получается дизъюнкция ложного и истинного высказывания, результат – истина.

Вывод: указанному условию удовлетворяют четыре записи приведенной базы данных, значит, верен вариант 4.

Ответ: 4.

Задача 3

База данных «Библиотека» состоит из трех связанных таблиц (табл. 9.3). Определите, сколько раз жители ул.

Балакирева брали в библиотеке книги Ирвина Шоу?

Таблица 9.3 Таблица читателей

№ п/п	Фамилия, Имя, Отчество	Адрес	Номер читательского билета
1	Мирошников Сергей Петрович	Балакирева ул., д. 25 кв. 20	А112703
2	Петренко Галина Юрьевна	Мира ул., д. 18, кв. 15	В514891
3	Агафонова Мария Борисовна	Ясная ул., д. 3, кв. 25	Б312187
4	Барашников Михаил Ильич	Балакирева ул., д. 8 кв. 12	А220157
5	Лымарь Артур Леонидович	Крайняя ул., д. 7, кв. 51	В612831
6	Комарова Лилия Сергеевна	Мира ул., д. 23, кв. 89	А340280

Таблица книг

Инв. номер	Автор	Название	Год издания
56714	Шоу И.	Вечер в Византии	2013
35214	Шоу И.	Любовь на темной улице	2013
87561	Булгаков М.А.	Собачье сердце	2010
54032	Булгаков М.А.	Мастер и Маргарита	2013
20004	Гоголь Н. В.	Мертвые души	2015
75020	Шоу И.	Богач, бедняк	2014

Таблица выдачи книг

Инв. номер книги	Номер читательского билета	Дата выдачи
56714	А112703	15.01.2015
20004	Б312187	23.01.2015
35214	А112703	05.02.2015
56714	А220157	10.03.2015
87561	А220157	23.03.2015
54032	В514891	08.02.2015

56714	Б312187	15.04.2015
75020	А340280	03.02.2015
20004	А112703	01.03.2015

Решение

1. По первой таблице определим номера читательских билетов жителей ул. Балакирева. Это номера А112703 и А220157.

2. Используя третью таблицу, замечаем, что жители ул. Балакирева брали книги с номерами 56714,56714, 35214, 87561 и 20004.

3. Из них книгами Ирвина Шоу являются книги с номерами 56714 и 35214. Книгу 56714 брали два раза.

Ответ: 3 книги.

9.2. Тестовые задания

1. Информационные системы – это:

1) Большие массивы данных об объектах и явлениях реального мира.

2) Программно-аппаратные средства для обработки информации об объектах и явлениях реального мира.

3) Большие массивы данных об объектах и явлениях реального мира и программно-аппаратные средства для их обработки.

4) Системы манипулирования данными.

2. Укажите наиболее точный аналог реляционной базы данных:

1) Неструктурированное множество данных.
2) Списки однородных данных.
3) Генеалогическое дерево.
4) Двумерная таблица.

3. В реляционных базах данных используются:

1) Несвязанные между собой таблицы.
2) Одна таблица, содержащая все данные.
3) Таблицы, между которыми устанавливаются связи.
4) Списки однородных данных.

4. Для чего предназначен объект «таблица»?

1) Для хранения данных.
2) Для архивирования данных.

3) Для ввода и удаления данных.
4) Для выборки данных.

5. Для чего предназначен объект «форма»?
1) Для хранения данных.
2) Для автоматического выполнения группы команд.
3) Для ввода данных базы и их просмотра.
4) Для выборки данных.

6. Для чего предназначен объект «запрос»?
1) Для ввода данных базы и их просмотра.
2) Для выборки и обработки данных.
3) Для хранения данных.
4) Для удаления данных из базы.

7. В чем заключается особенность типа данных «четчик»?
1) Служит для ввода целых и действительных чисел.
2) Имеет свойство автоматически увеличиваться.
3) Имеет свойство автоматического пересчета при удалении записи.
4) Служит для ввода шифров.

8. Первичный ключ таблицы – это:
1) Номер первой по порядку записи.
2) Любое поле числового типа.
3) Одно или несколько полей, значения которых однозначно определяют любую запись в таблице.
4) Первое поле числового типа.

9. Какое поле можно считать уникальным?
1) поле, значения в котором не могут повторяться;
2) поле, которое носит уникальное имя;
3) поле, значение которого имеют свойства наращивания;
4) поле, все значение которого одинаковые.

10. В базе данных

Фамилия	Имя	Отчество	Оклад
Зайцев	Семен	Петрович	40000
Петров	Сергей	Сергеевич	45000
Сидорин	Тимофей	Михайлович	30500
Опарин	Филипп	Алексеевич	25000

записи упорядочены по полю:
1) Имя 2) Фамилия 3) Отчество 4) Оклад

11. Сколько записей в табл. 9.1 удовлетворяют условию запроса Часть А>= 9 и Часть В>= 9 и Часть С > 7 :
1) 5 2) 3 3) 2 4) 1

12. Запросу (Химия = 5 илиБиология = 5) иРусский = 5 и Первая_буква (Фамилия) = "Д" в базе данных:

Номер	Фамилия	Имя	Русский	Химия	Биология
1	Дронов	Павел	4	4	5
2	Капитонов	Олег	5	5	5
3	Донец	Сергей	5	4	5
4	Донцов	Михаил	5	5	4

соответствуют записи:
1) 1, 3, 4 2) 2, 4 3) 1, 4 4) 3, 4

13. Сколько записей в нижеследующем фрагменте турнирной таблицы удовлетворяют условию «Место <=4 И (Н>2 ИЛИ О>6)»:

Место	Участник	В	Н	П	О
1	Силин	5	3	1	6 ½
2	Клеменс	6	0	3	6
3	Холево	5	1	4	5 ½
4	Яшвили	3	5	1	5 ½
5	Бергер	3	3	3	4 ½
6	Численко	3	2	4	4

1) 2 2) 5 3) 4 4) 6

14. Запрос к базе данных с полями Фамилия, Год рождения, Класс, Оценка для вывода списка студентов 1-го курса 1999 года рождения, имеющих оценки 4 или 5, содержит выражение:
1) Курс > 1 и Оценка = 4 и Год_рождения = 1999
2) Курс=1 или Оценка > 4 или Год_рождения = 1999
3) Оценка >= 4 и Год_рождения = 1999 и Курс = 1

4) Оценка >= 4 и Год_рождения>1999 или Курс=1

15. Ответы

Вопрос	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
Ответ	3	4	3	1	3	2	2	3	1	1	3	4	1	3

10.3. Лабораторная работа

Работа с базой данных

Цель: научиться создавать связи между таблицами, научиться создавать запросы на выборку данных из таблиц.

Задание 1

Свяжите таблицы «Водители», «Автомобильный фонд» и «Учет» между собой с целью дальнейшего получения данных из этих таблиц.

Порядок выполнения задания

1. Запустите Microsoft Access 2007 (Пуск → Программы → Microsoft Access).

2. Для создания связей между таблицами сначала их закроем, а затем на панели инструментов «Работа с данными» нажмем кнопку «схема данных». Появится окно «Схема данных».

3. В группе команд «Связи» нажимаем кнопку «Отобразить таблицу». Откроется окно «Добавление таблицы». Добавить поочередно все три табицы: «Автомобильный фонд», «Учет» и «Водители». В рабочей области отобразятся эти пока еще не связанные таблицы.

4. Свяжем таблицы «Автомобильный фонд» и «Учет». Для этого надо связать поле «Код» в таблице «Автомобильный фонд» с полем «Код книги» в таблице «Учет».

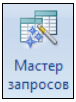
5. На вкладке «Конструктор» нажать кнопку «Изменить связи». Появится окно «Изменение связей», в котором отображаются названия таблиц и связываемых полей. Нажать кнопку «Новое». Появится окно «Создание». Для создания связей в левой и правой частях окна в соответствующих полях надо указать названия таблиц и связываемых полей и нажать «ОК». Появится окно «Изменение связей», в котором отображаются названия таблиц и связываемых полей, поставить галочку «Обеспечение целостности данных» и нажать кнопку «Создать».

6. Аналогично связать таблицы «Учет» и «Водители».

Задание 2

Создать запрос на выборку, содержащий следующие поля: ФИО и номер автомобиля из таблицы «Водители», название из таблицы «Автомобильный фонд», автор из таблицы «Учет» и дата выдачи. Поле «Дата выдачи» сортировать по возрастанию.

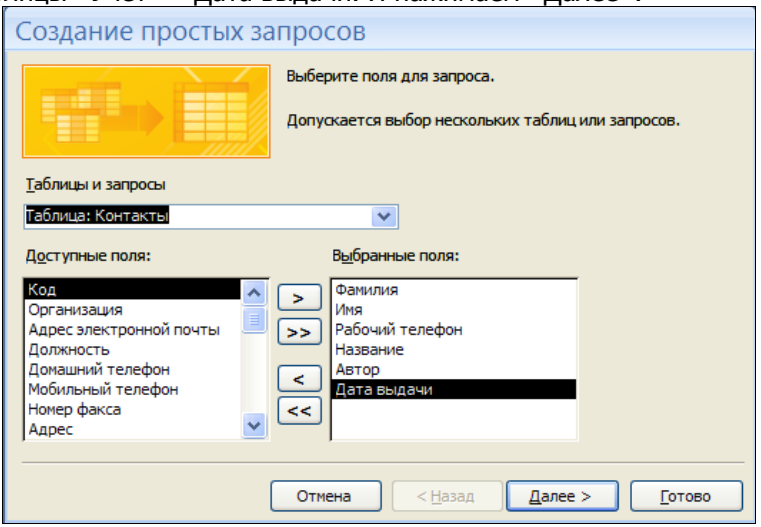
Порядок выполнения задания

1. Переходим на вкладку «Создание» и нажимаем кнопку 

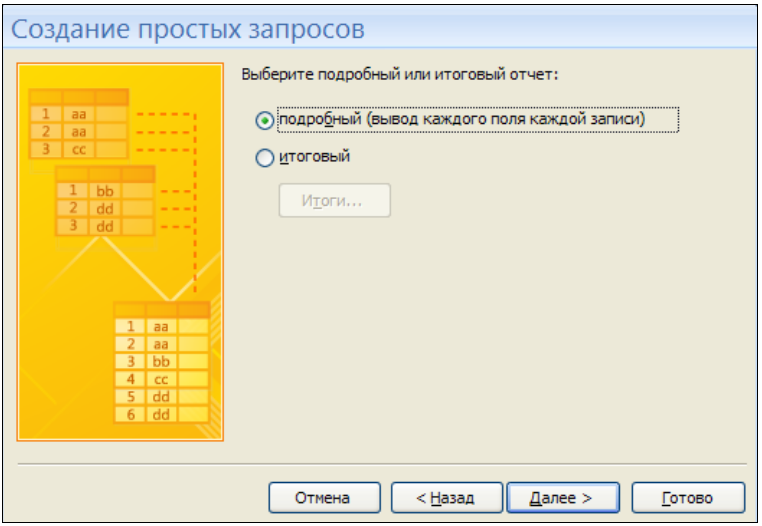
.

2. Выбираем «Простой запрос» и нажимаем ОК.

3. В раскрывающемся списке «Таблицы и запросы» выбираем таблицу «Водители», из списка «Доступные поля» выбираем: Фамилия, Имя и номер автомобиля. Затем из таблицы «Автомобильный фонд» выбираем: Название и Автор; а из таблицы «Учет» – Дата выдачи. И нажимаем «Далее».



4. Выбираем подробный отчет и нажимаем «Далее».



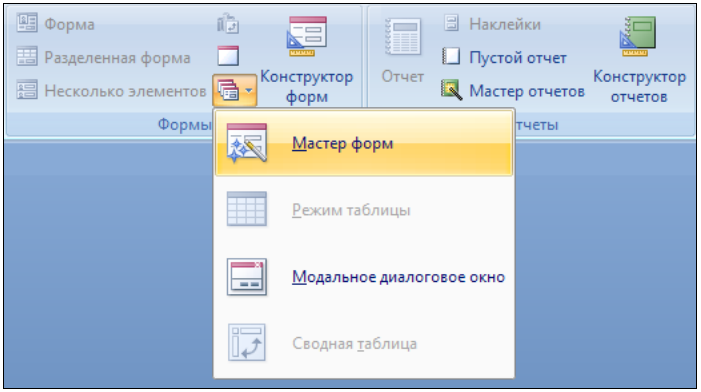
5. Вводим имя запроса «Выдача автомобиля», ставим переключатель на «Открыть запрос для просмотра данных» и нажимаем «Готово».

Задание 3

Создать форму для добавления ввода данных в таблицу «Водители», используя «Мастер форм».

Порядок выполнения задания

1. Переходим на закладку «Создание» и запускаем «Мастер форм».



2. В списке «Таблицы и запросы» выберите Таблица: Водители.

3. Кликом на кнопке «>>»добавьте все поля в список выбранных полей.

4. Выделите поле «Код» и кликом на кнопке « < » уберите из списка выбранных полей, так как значение этого поля является кодом, оно неважно для пользователя и поэтому его не следует включать в форму.

5. Кликните на кнопке «Далее >».

6. В следующем окне выберите внешний вид формы – выровненный и кликните на кнопке «Далее > ».

7. В третьем окне мастера выберите стиль оформления формы (какая понравится) и нажмите на кнопку «Далее > ».

8. В четвертом окне нажмите на кнопку «Готово».

ИСПОЛЬЗУЕМАЯ ЛИТЕРАТУРА

1. Батлер Э. Microsoft Office Access 2007: профессиональное программирование / Э. Батлер. – М.: Вильямс 2009.

2. Гальченко Г.А. Информатика для колледжей: учеб. пособие / Г.А. Гальченко, О.Н. Дроздова. – М.: Феникс, 2017. – 320 с.

3. Гальченко Г.А. Информационные технологии в организации перевозочных услуг / Г.А. Гальченко, А.В. Алейникова // Технология транспортных процессов на Дону. – Новочеркасск: Изд-во Лик, 2016. – Стр. 124–126.

4. Гальченко Г.А. Использование информационных технологий для повышения качества образования в области организации транспортных процессов / Г.А. Гальченко, О.Н. Дроздова // Инновационные технологии в науке и образовании ИНТО-2015: сб. материалов Междунар. науч.-метод. конф. посвященной 85 ДГТУ. – Ростов н/Д: Издательский центр ДГТУ, 2015

5. Гальченко Г.А. Нанотехнологии и довузовская подготовка / Г.А. Гальченко, В.И. Логвинов, А.А. Тихонов // Вестн. Донск. гос. техн. ун-та. – 2008. – Т. 8. – № 1 (36). – С. 101–105.

6. Гальченко Г.А. Обработка информации в транспортных задачах как фактор формирования исследовательской компетенции студентов / Г.А. Гальченко, В.И. Логвинов // Перспективы развития информационных технологий: сб. науч. тр. VI Междунар. науч-практ. конф. – Новосибирск, 2013.

7. Гальченко Г.А. Применение метода активного обучения на базе компьютерной модели «Мультибрендовый автосалон» / Г.А. Гальченко, Н.С. Донцов, А.А. Сарабашев // Символ науки. – Новосибирск. – 2016. – № 2. –Ч.1. – С. 90–96.

8. Гальченко Г.А. Расчет основных характеристик транспортного потока на участке ул. Орбитальная – ул. Беляева г. Ростов-на-Дону / Г.А. Гальченко, О.Н. Дроздова, А.А. Детистова // Безопасность, дорога, дети: практика, опыт, перспективы и технологии материалы форума. – Ростов-на-Дону, 2015. – С. 138–141.

9. Гальченко Г.А. Элементы оптимизации транспортных процессов как фактор формирования практической компетентности студентов / Г.А. Гальченко, В.И. Логвинов // Наукоемкие технологии и инновации ; Белгородский государственный технологический университет им. В.Г. Шухова. – 2014. – С. 144–149.

10.Джексон Г. Проектирование реляционных баз данных для использования с микроЭВМ / Г. Джексон; пер. с англ. – М.: Мир, 1991. – 252 с.

11.Диагностирование и испытание электрооборудования транспортных машин: учеб. пособие / С.И. Попов, В.Ю. Валявин, С.Ф. Подуст и др. – Ростов н/Д: Издательский центр ДГТУ, 2010. – 115 с.

12.Иванов В.В. Вибрационное механохимическое цинкование крепёжных деталей автомобиля / В.В. Иванов, Г.А. Гальченко // Автомобильная промышленность. – 2016. – № 2. – С. 30–32.

13.Иванов В.В. Программный комплекс t-flex технология 10 / В.В. Иванов // САПР и графика. – 2006. – № 9. – С. 44.

14.Кириллов В.В. Введение в реляционные базы данных / В.В. Кириллов, Г.Ю. Громов. – М.: BHV, 2009.

15.Коннолли Т. Базы данных: проектирование, реализация и сопровождение. Теория и практика: учеб. пособие / Т. Коннолли, К. Бегг, А. Страчан; пер. с англ. – 2-е изд.– М.: Вильямс, 2000. – 1120 с.

16.Марченко Э.В. Метод нанесения твердосмазочных материалов на стальной канат в процессе его производства / Э.В. Марченко, С.И. Попов, Ю.В. Марченко и др. // Виброволновые процессы в технологии обработки деталей высокотехнологичных изделий: сб. тр. Междунар. науч. симпозиума технологов-машиностроителей (3–6 октября). – Ростов н/Д, 2017. – С. 131–134.

17.Марченко Э.В. Устройство по нанесению твердосмазочных

материалов на стальной канат в процессе его производства / Э.В. Марченко, С.И. Попов, Ю.В. Марченко // Научно-технический прогресс: актуальные и перспективные направления будущего: сб. материалов VI Междунар. науч.-практ. конф. (18 августа). Т. II. – Кемерово: ЗапСибНЦ, 2017.– С. 111–114.

18.Плахова В.Г. Формирование математической компетенции у будущих инженеров / В.Г. Плахова // Актуальные проблемы математики и методики преподавания математики: сб. науч. тр. / под ред. С.Н. Дорофеева. – Пенза: ПГТА, 2007.

19.Попов С.И. Разработка интернет-курса по дисциплине «Техническая эксплуатация автомобилей» / С.И. Попов, Ю.В. Марченко, Н.С. Донцов // Аспекты развития науки, образования и модернизации промышленности: материалы Всерос. науч.-практ. конф. с междунар. участием (20–21 апреля, Таганрог). – Ростов н/Д. – 2017. – С. 61–63.

20.Скудина А.А. Анализ эффективности работы общественного городского транспорта с учетом мнений жителей г. Ростова-на-Дону / А.А. Скудина, А.Г. Исаев, Э.В. Марченко // Научно-технический прогресс: актуальные и перспективные направления будущего: сб. материалов V Междунар. науч.-практ. конф. Западно-Сибирский научный центр. – 2017. – С. 181–183.

21.Скудина А.А. Повышение уровня удобства движения по средствам передачи информации между автомобилями / А.А. Скудина, А.Ю. Чумакова // Безопасность, дорога, дети: практика, опыт, перспективы и технологии. – Ростов н/Д. – 2015. – С. 233–235.

22.Сокол Н.А. Основы конструкции и расчета автомобиля: учеб. / Н.А. Сокол, С.И. Попов. – Ростов н/Д: Феникс, 2006. – 303 с.

23.Технические измерения на транспорте: учеб. пособие / Э.В. Марченко, С.И. Попов, Ю.В. Марченко и др.; Донской гос. техн. ун-т. – Ростов н/Д: ДГТУ, 2017. – 81 с.

24.Хомоненко А.Д. Базы данных: учеб. для высших учебных заведений / А.Д. Хомоненко, В.М. Цыганков, М.Г. Мальцев; под ред. проф. А.Д. Хомоненко. – Изд. 2-е, доп. и перераб. – СПб.: КОРОНА принт, 2002. – 672 с.

25.http://www.evolkov.net/case/case.study.html

26.http://www.magistr.net.ua

27.Ivanov V.V., Popov S.I., Kirichek A.V. Qualitative Characteristics of MoS_2 Solid-Lubricant Coating Formed by Vibro-Wave Impact of Free-Moving Indenters // Key Engineering Materials, Vol. 736, pp. 18-22, 2017 DOI 10.4028/www.scientific.net/KEM.736.18