Министерство образования Красноярского края

Краевое государственное бюджетное профессиональное образовательное

учреждение «Красноярский колледж радиоэлектроники и информационных технологий»



**МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ**

По профессиональному модулю \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ОП.08 Основы проектирования баз

интеллектуальных интегрированных систем

Курс \_\_\_\_

2

Для специальности (код и наименование)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

09.02.08 Интеллектуальные интегрированные системы

Красноярск, 2025

Методические рекомендации составлены:

Преподавателем высшей категории КГБПОУ СПО «ККРИТ» Е.В. Харитоновой

Преподавателем высшей категории КГБПОУ СПО «ККРИТ» Л.В. Шайхутдиновой

Преподавателем первой категории КГБПОУ СПО «ККРИТ» К.В. Савельевой

РАССМОТРЕНО

на заседании цикловой комиссии преподавателей

профессионального цикла информационно-технического профиля

Протокол №\_\_\_ от «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2025 г.

Председатель ЦК \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.В. Харитонова

Ответственный редактор: зам. директора по учебной работе М.А. Полютова

Одобрено Методическим советом КГБПОУ СПО «ККРИТ»

протокол № \_\_ от «\_\_\_» \_\_\_\_\_\_ 2025 г.

Председатель методического совета

Зам. директора по УР М.А. Полютова

Содержание

Пояснительная записка

Перечень практических работ

Методические указания для студентов по выполнению практических занятий

## ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Методические указания по выполнению практических занятий по предмету ОП.08 Основы проектирования баз данных разработаны с целью оказания помощи студентам спе- циальности 09.02.08 Интеллектуальные интегрированные системы и преподавателям по ор- ганизации практических занятий по изучаемой дисциплине, в соответствии с требованиями федерального государственного стандарта среднего профессионального образования.

Методические разработка включает в себя краткие теоретические сведения, указа- ния по выполнению практических работ, контрольные вопросы, формы контроля.

Программой учебной дисциплины ОП.08 Основы проектирования баз данных предусмотрено проведение практических занятий в количестве **32 часа** по специальности:

09.02.08 Интеллектуальные интегрированные системы.

В результате освоения учебной дисциплины обучающийся **должен: уметь:**

* проектировать реляционную базу данных;
* использовать язык запросов для программного извлечения сведений из баз дан-

ных

* устанавливать систему управления базами данных (СУБД);
* выполнять регламентные процедуры по резервированию данных. В результате освоения учебной дисциплины обучающийся **должен знать:**
* основы теории баз данных;
* модели данных;
* особенности реляционной модели и проектирование баз данных;
* принципы проектирования баз данных;
* обеспечение непротиворечивости и целостности данных;
* средства проектирования структур баз данных язык запросов SQL.

В результате освоения учебной дисциплины ОП.08 Основы проектирования баз дан-

ных у выпускника должны быть сформированы следующие компетенции:

ОК 01. Выбирать способы решения задач профессиональной деятельности, приме- нительно к различным контекстам.

ОК 02. Использовать современные средства поиска, анализа и интерпретации инфор- мации, и информационные технологии для выполнения задач профессиональной деятель- ности.

ОК 04. Эффективно взаимодействовать и работать в коллективе и команде

ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста. ОК 09. Пользоваться профессиональной документацией на государственном и ино-

странном языках.

ПК 2.1. Осуществлять мониторинг функционирования интеграционного решения ПК 2.2. Выполнять работы по документированию функций системы

Характерная черта практических занятий – индивидуальное выполнение заданий, са- мостоятельное приобретение знаний. В связи с этим предусмотрены работы по всем основ- ным разделам курса. Перед выполнением практической работы обучающийся получает опе- режающее теоретическое домашнее задание. На занятии объясняются вопросы, уточняются определения, которые помогают выполнению заданий. Обучающийся может просмотреть запись объяснения любой примерной работы по всем темам. И только после этого обучаю- щийся приступает к выполнению практической работы.

При выполнении работы обучающийся должен самостоятельно изучить методиче- ские рекомендации по проведению практической работы, подготовить ответы на контроль- ные вопросы. Все практические задания выполняются за компьютером, теоретические во- просы сдаются устно или письменно.

|  |  |
| --- | --- |
| **№ п/п** | **НАЗВАНИЕ ПРАКТИЧЕСКИХ ЗАНЯТИЙ** |
| **1.** | **Практическое занятие № 1.** Решение задач по теме: Реляционная алгебра |
| **2.** | [**Практическая работа № 2.** Преобразование реляционной БД в сущности и связи](#_bookmark0) |
| **3.** | [**Практическая работа № 3.** Нормализация реляционной БД, освоение принци-](#_bookmark0) [повпроектирования БД](#_bookmark0) |
| **4.** | [**Практическая работа № 4.** Проектирование реляционной БД. Нормализация таб-](#_bookmark1) [лиц](#_bookmark1). Задание ключей. Создание основных объектов БД |
| **5.** | [**Практическая работа № 5.** Создание проекта БД. Создание БД. Редактирование и](#_bookmark2) [модификация таблиц](#_bookmark2) |
| **6.** | [**Практическая работа № 6.** Создание ключевых полей. Задание индексов. Установ-](#_bookmark3) [ление иудаление связей между таблицами](#_bookmark3) |
| **7.** | [**Практическая работа № 7.** Редактирование, добавление и удаление записей в таб-](#_bookmark4) [лице.](#_bookmark4) [Применение логических условий к записям. Открытие, редактирование и по-](#_bookmark4) [полнение](#_bookmark4) [табличного файла](#_bookmark4) |
| **8.** | [**Практическая работа № 8.** Проведение сортировки и фильтрации данных. Поиск](#_bookmark5) [данных](#_bookmark5) [по одному и нескольким полям. Поиск данных в таблице](#_bookmark5) |
| **9.** | [**Практическая работа № 9.** Работа с переменными. Написание программного файла](#_bookmark6) [и](#_bookmark6) [работа с табличными файлами. Заполнение массива из табличного файла. Запол-](#_bookmark6) [нение](#_bookmark6) [табличного файла из массива](#_bookmark6) |
| **10.** | [**Практическая работа № 10.** Создание формы. Управление внешним видом формы](#_bookmark7) |
| **11.** | [**Практическая работа № 11.** Добавление записей в табличный файл из двумерного](#_bookmark8) [массива. Работа с командами ввода-вывода. Использование функций для работы с](#_bookmark8) [массивами](#_bookmark8) |
| **12.** | [**Практическая работа № 12.** Создание меню различных видов. Модификация и](#_bookmark9) [управление меню](#_bookmark9). [Создание рабочих и системных окон. Добавление элементов](#_bookmark10) [управления рабочим окном](#_bookmark10) |
| **13.** | [**Практическая работа № 13.** Создание файла проекта базы данных. Создание ин-](#_bookmark11) [терфейса](#_bookmark11) [входной формы. Использование исполняемого файла проекта БД, приемы](#_bookmark11) [создания и](#_bookmark11) [управления](#_bookmark11) |
| **14.** | [**Практическая работа № 14.** Задание значений и ограничений поля. Проверка вве-](#_bookmark12) [денного](#_bookmark12) [в поле значения. Отображение данных числового типа и типа дата](#_bookmark12) |
| **15.** | [**Практическая работа № 15.** Создание и модификация таблиц БД. Выборка данных](#_bookmark13) [из БД.Модификация содержимого БД](#_bookmark13) |
| **16.** | [**Практическая работа № 16.** Обработка транзакций. Использование функций за-](#_bookmark14) [щиты для](#_bookmark14) [БД](#_bookmark14) |

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ СТУДЕНТОВ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ**

**гебра»**

**Практическая работа №1. Решение задач по теме: «Реляционная ал-**

**Цель занятия:** Изучение основ реляционной алгебры.

В основе реляционной алгебры лежит идея о том, что так как отношение – это мно-

жество кортежей, то и средства манипулирования отношениями должны быть такими же, как традиционные теоретико–множественные операции, дополненные специфическими для баз данных операциями.

Расширенный начальный вариант алгебры, определённый Коддом, состоит из восьми алгебраических операций, которые делятся на два класса – теоретико–множествен- ные операции и специальные реляционные операции.

В состав теоретико–множественных операций входят:

* Объединение: возвращает отношение, содержащее все кортежи, которые принад- лежат либо одному, либо обоим из заданных отношений;
* Пересечение: возвращает отношение, содержащее кортежи, которые принадле- жат одновременно двум заданным отношениям;
* Разность: возвращает отношение. содержащее все кортежи, которые принадле- жат первому из двух заданных отношений и не принадлежат второму;
* Декартово произведение: возвращает отношение, содержащее все возможные кортежи, которые являются сочетанием двух кортежей, принадлежащих соответственно двум заданным отношениям;

произведение отношений.

Специальные реляционные операции включают:

* Проекция: возвращает отношение, содержащее все кортежи заданного отноше- ния, которые остались в этом отношении после исключения из него некоторых атрибутов;
* Соединение: возвращает отношение, кортежи которого – это сочетания двух кор- тежей (принадлежащих соответственно двум определенным отношениям), имеющих общее значение для одного или нескольких общих атрибутов этих двух отношений (и такие общие значения в результирующем кортеже появляются только один раз, а не дважды).
* Деление: для двух отношений, бинарного и унарного, возвращает отношение, со- держащее все значения одного атрибута бинарного отношения, которые соответствуют (в другом атрибуте) всем значениям в унарном отношении.
* Выборка: возвращает отношение, содержащее кортежи из заданного отношения, которые удовлетворяют указанным условиям;

Кроме того, в состав алгебры включается операция присваивания, позволяющая со- хранить в базе данных результаты вычисления алгебраических выражений, и операция пе- реименования атрибутов, дающая возможность корректно сформировать заголовок (схему) результирующего отношения:

* Операция переименования производит отношение, тело которого совпадает с телом операнда, но имена атрибутов изменены.
* Операция присваивания позволяет сохранить результат вычисления реляцион- ного выражения в существующем отношении БД.

Поскольку результатом любой реляционной операции (кроме операции присваива- ния) является некоторое отношение, можно образовывать реляционные выражения, в кото- рых вместо отношения-операнда некоторой реляционной операции находится вложенное реляционное выражение.

Для алгебры, у которой операции замкнуты относительно понятия отношения, каж- дая операция должна производить отношение с двумя составляющими: телом и заголовком. Только в этом случае будет действительно возможно строить вложенные выражения. Опе-

рации объединения, пересечения взятия разности (вычитания) требуют от отношений-опе- рандов совместимых по типу. Два отношения называются совместимыми по типу, если каж- дое из них имеет одинаковое множество имен атрибутов и если соответствующие атрибуты определены на одном и том же домене. Если необходимо выполнить операцию объедине- ния, пересечения или вычитания двух отношений, которые почти совместимы по типу, за исключением некоторых различий в именах атрибутов, можно использовать оператор пе- реименования, чтобы сделать эти отношения полностью совместимыми по типу, прежде чем выполнить необходимую операцию.

Примеры использования реляционной алгебры для выражения словесных запросов в виде формулы

Дана БД предприятия, состоящая из трех таблиц:

|  |  |  |  |
| --- | --- | --- | --- |
| **ПОСТАВЩИК** | | | |
| **ПНОМ** | **ПФАМ** | **СТАТУС** | **ГОРОД** |
| П1 | Иванов | 20 | Воронеж |
| П2 | Петров | 15 | Москва |
| П3 | Сидоров | 10 | Москва |
| П4 | Зайцев | 30 | Воронеж |
| П5 | Волков | 20 | Киев |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ДЕТАЛЬ** | | | | |
| **ДНОМ** | **ДНАЗВ** | **ЦВЕТ** | **ВЕС** | **ГОРОД** |
| Д1 | Гайка | Красный | 12 | Воронеж |
| Д2 | Болт | Зеленый | 17 | Москва |
| Д3 | Шайба | Голубой | 17 | Минск |
| Д4 | Шайба | Красный | 14 | Воронеж |
| Д5 | Шуруп | Голубой | 12 | Москва |
| Д6 | Гвоздь | Красный | 19 | Воронеж |

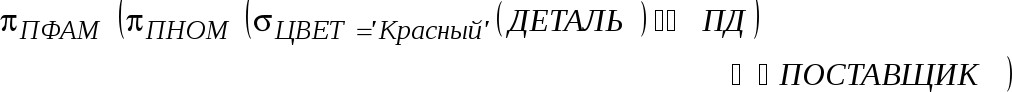
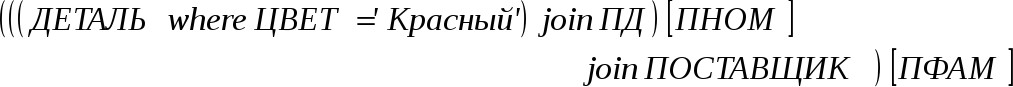
|  |  |  |
| --- | --- | --- |
| **ПД** |  |  |
| **ПНОМ** | **ДНОМ** | **ШТ** |
| П1 | Д1 | 300 |
| П1 | Д2 | 200 |
| П1 | Д3 | 400 |
| П1 | Д4 | 200 |
| П1 | Д5 | 100 |
| П1 | Д6 | 100 |
| П2 | Д1 | 300 |
| П2 | Д2 | 400 |
| П3 | Д3 | 200 |
| П4 | Д2 | 200 |
| П4 | Д4 | 300 |
| П4 | Д5 | 400 |

Построить выражения реляционной алгебры, которые позволят получить ответ на вопросы:

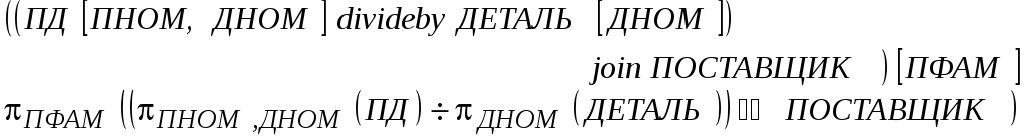
1. C:\Users\student\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B9316B45.tmpПолучить имена поставщиков, которые поставляют деталь Д2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ПНОМ** | **ДНОМ** | **ШТ** | **ПФАМ** | **СТАТУС** | **ГОРОД** |
| П1 | Д1 | 300 | Иванов | 20 | Воронеж |
| П1 | Д2 | 200 | Иванов | 20 | Воронеж |
| П1 | Д3 | 400 | Иванов | 20 | Воронеж |
| П1 | Д4 | 200 | Иванов | 20 | Воронеж |
| П1 | Д5 | 100 | Иванов | 20 | Воронеж |
| П1 | Д6 | 100 | Иванов | 20 | Воронеж |
| П2 | Д1 | 300 | Петров | 15 | Москва |
| П2 | Д2 | 400 | Петров | 15 | Москва |
| П3 | Д3 | 200 | Сидоров | 10 | Москва |
| П4 | Д2 | 200 | Зайцев | 30 | Воронеж |
| П4 | Д4 | 300 | Зайцев | 30 | Воронеж |
| П4 | Д5 | 400 | Зайцев | 30 | Воронеж |

1. Получить имена поставщиков, которые поставляют по крайней мере одну красную деталь



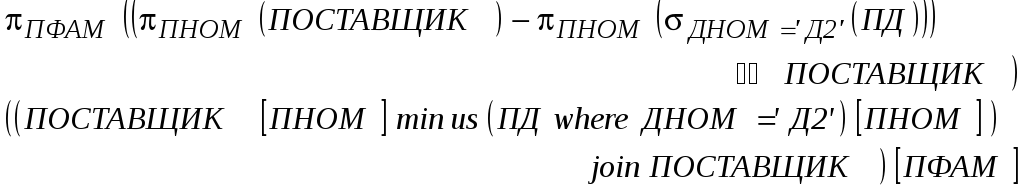
1. Получить имена поставщиков, которые поставляют все детали



1. Получить номера поставщиков, поставляющих по крайней мере все те де- тали, которые поставляет поставщик П2

C:\Users\student\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\9A1F504F.tmp

1. Получить имена поставщиков, которые не поставляют деталь Д2

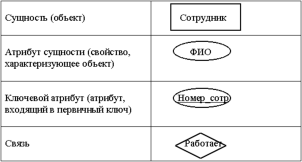


**Практическая работа № 2 Преобразование реляционной БД в сущности и связи Цель работы:** выработать практические навыки моделирования предметной обла-

сти и построении ER-модели данных, закрепить технологию проектирования БД, закрепить основные понятия теории реляционных баз данных, освоить технологию построения ER- диаграмм, научиться получать реляционные БД из ER-диаграмм

## Сущность-связь

Работа с базой данных начинается с построения модели. Наиболее распространенной является ER-модель (entity-relationship model) - модель "Сущность-связь". Для "ручного" построения ER-модели на практике будем использовать простую систему обозначений, предложенную Питером Ченом (обозначения, встречающиеся в разных источниках, могут



отличаться от нижеприведенных):

Рисунок 1.1. Сущность - связь

Первичный ключ - атрибут или группа атрибутов, однозначно идентифицирующих объект. Первичный ключ может состоять из нескольких атрибутов, тогда подчеркивается каждый из них.

Связи между объектами могут быть 3-х типов: *Один - к одному*. Этот тип связи озна- чает, что каждому объекту первого вида соответствует не более одного объекта второго вида, и наоборот. Например: сотрудник может руководить только одним отделом, и у каж- дого отдела есть только один руководитель. *Один - ко многим.* Этот тип связи означает, что каждому объекту первого вида может соответствовать более одного объекта второго вида, но каждому объекту второго вида соответствует не более одного объекта первого вида. Например: в каждом отделе может быть множество сотрудников, но каждый сотрудник ра- ботает только в одном отделе. *Многие - ко многим.* Этот тип связи означает, что каждому объекту первого вида может соответствовать более одного объекта второго вида, и наобо- рот. Например: каждый счет может включать множество товаров, и каждый товар может входить в разные счета.

## Реляционная структура данных

В конце 60-х годов появились работы, в которых обсуждались возможности приме- нения различных табличных даталогических моделей данных. Наиболее значительной из них была статья сотрудника фирмы IBM д-ра Эдварда Кодда (Codd E.F., A Relational Model of Data for Large Shared Data Banks. CACM 13: 6, June 1970), где впервые был применен термин «реляционная модель данных».

Будучи математиком по образованию, Э. Кодд предложил использовать для обра- ботки данных аппарат теории множеств (объединение, пересечение, разность, декартово произведение). Он показал, что любое представление данных сводится к совокупности дву- мерных таблиц особого вида, известного в математике как *отношение* – relation.

*Наименьшая единица данных реляционной модели – это отдельное атомарное (неразложи- мое) для данной модели значение данных.*

Так, в одной предметной области фамилия, имя и отчество могут рассматриваться как единое значение, а в другой – как три различных значения.*Доменом называется множе- ство атомарных значений одного и того же типа.* Так, на рис. домен пунктов отправления (назна- чения) – множество названий населенных пунктов, а домен номеров рейса – множество целых по- ложительных чисел.

Таблица 1.1. Рейс

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Номер рейса | Дни недели | Пункт отправления | Время вылета | Пункт назначения | Время прибытия | Тип самолета | Стоимость билета |
| 138 | 2\_4\_7 | Баку | 21.12 | Москва | 0.52 | ИЛ-86 | 115.00 |
| 57 | 3\_6 | Ереван | 7.20 | Киев | 9.25 | ТУ-154 | 92.00 |
| 1234 | 2\_6 | Казань | 22.40 | Баку | 23.50 | ТУ-134 | 73.50 |
| 242 | 1 по 7 | Киев | 14.10 | Москва | 16.15 | ТУ-154 | 57.00 |
| 86 | 2\_3\_5 | Минск | 10.50 | Сочи | 13.06 | ИЛ-86 | 78.50 |
| 137 | 1\_3\_6 | Москва | 15.17 | Баку | 18.44 | ИЛ-86 | 115.00 |
| 241 | 1 по 7 | Москва | 9.05 | Киев | 11.05 | ТУ-154 | 57.00 |
| 577 | 1\_3\_5 | Рига | 21.53 | Таллин | 22.57 | АН-24 | 21.50 |
| 78 | 3\_6 | Сочи | 18.25 | Баку | 20.12 | ТУ-134 | 44.00 |
| 578 | 2\_4\_6 | Таллин | 6.30 | Рига | 7.37 | АН-24 | 21.50 |

Смысл доменов состоит в следующем. Если значения двух атрибутов берутся из од- ного и того же домена, то, вероятно, имеют смысл сравнения, использующие эти два атри- бута (например, для организации транзитного рейса можно дать запрос «Выдать рейсы, в которых время вылета из Москвы в Сочи больше времени прибытия из Архангельска в Москву»). Если же значения двух атрибутов берутся из различных доменов, то их сравне- ние, вероятно, лишено смысла: стоит ли сравнивать номер рейса со стоимостью билета?

Отношение на доменах D1, D2, ..., Dn состоит из заголовка и тела. На рис. 3.4 приве- ден пример отношения для расписания движения самолетов (таблица 1). Ai - атрибуты, Vi

- значения атрибутов.

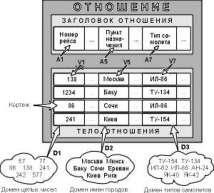


Рисунок 1.2. Отношение с математической точки зрения

*Заголовок отношения состоит из такого фиксированного множества атрибутов A1, A2,*

*..., An, что существует взаимно однозначное соответствие между этими атрибутами Ai и определяющими их доменами Di (i=1,2,...,n).*

*Тело отношения состоит из меняющегося во времени множества кортежей, где каждый кортеж состоит в свою очередь из множества пар атрибут-значение (Ai:Vi), (i=1,2,...,n), по одной такой паре для каждого атрибута Ai в заголовке.*

Для любой заданной пары атрибут-значение (Ai:Vi) Vi является значением из един- ственного домена Di, который связан с атрибутом Ai.

*Степень отношения – это число его атрибутов.*

Отношение степени один называют унарным, степени два – бинарным, степени три

–

тернарным, ..., а степени n – n-арным. Степень отношения «Рейс» (таблица 1) равна

8.

*Кардинальное число или мощность отношения – это число его кортежей.*

Мощность отношения «Рейс» равна 10. Кардинальное число отношения изменяется

во времени в отличие от его степени.

Поскольку отношение – это множество, а множества по определению не содержат совпадающих элементов, то никакие два кортежа отношения не могут быть дубликатами друг друга в любой произвольно заданный момент времени.

*Пусть R – отношение с атрибутами A1, A2, ..., An. Говорят, что множество атрибутов K=(Ai, Aj, ..., Ak) отношения R является возможным ключом R тогда и только тогда, когда удо- влетворяются два независимых от времени условия*:

Уникальность: в произвольный заданный момент времени никакие два различных кортежа R не имеют одного и того же значения для Ai, Aj, ..., Ak.

Минимальность: ни один из атрибутов Ai, Aj, ..., Ak не может быть исключен из K без нарушения уникальности.

Каждое отношение обладает хотя бы одним возможным ключом, поскольку по мень- шей мере комбинация всех его атрибутов удовлетворяет условию уникальности. Один из возможных ключей (выбранный произвольным образом) принимается за его первичный ключ. Остальные возможные ключи, если они есть, называются альтернативными ключами. Вышеупомянутые и некоторые другие математические понятия явились теоретиче-

ской базой для создания реляционных СУБД, разработки соответствующих языковых средств и программных систем, обеспечивающих их высокую производительность, и созда- ния основ теории проектирования баз данных.

Также на практике широко используются неформальные эквиваленты этих понятий: Отношение – Таблица, Кортеж – Строка таблицы или Запись, Атрибут – Столбец Таблицы или Поле.

При этом принимается, что «запись» означает «экземпляр записи», а «поле» означает

«имя и тип поля».

Задание для практической работы

## Задание 1. Проектирование реляционных Баз Данных.

Вариант 1.

Построить реляционную таблицу Базы Данных имен родственников студентов вашей группы, содержащую данные об именах родителей, братьев и сестер студентов. Вариант 2.

Построить реляционную таблицу Базы Данных домов, где живут студенты вашей группы, содержащую данные о районе расположения дома, количестве этажей в нем и но- мере этажа, где живет студент.

Вариант 3.

Построить реляционную таблицу Базы Данных дней рождения студентов вашей группы, содержащую данные о дате рождения, знаке зодиака и годе по Китайскому кален- дарю.

Вариант 4.

Построить реляционную таблицу Базы Данных Сотовых телефонов студентов вашей группы, содержащую данные о модели телефона, типе корпуса, операторе.

Технология выполнения работы и оформление отчета

* 1. Придумайте заголовок отношения и запишите его в отчет.
  2. Определите атрибуты отношения. Начертите сетку таблицы в отчет и занесите в нее атрибуты.
  3. Опросите студентов вашей группы и занесите полученные данные в таблицу.
  4. На чертеже таблицы укажите чему соответствуют понятия: Заголовок отношения, тело отношения, атрибут отношения, кортеж отношения.
  5. Определите и запишите в отчет степень отношения и мощность отношения.
  6. Дайте определение первичного ключа. Укажите Первичный ключ получивше- гося отношения
  7. Докажите, что у вас получилась реляционная таблица, для этого укажите типы данных всех атрибутов.

**Задание 2. Проектирование Баз Данных. ER-диаграммы.**

Формулировка задания. По описанию предметной области построить логическую модель БД методом ER-диаграмм, на основании которой построить набор таблиц БД. Вариант 1.

Описание предметной области (Ресторан).

Посетители ресторана обслуживаются за столиками. За одним столом может распо- лагаться не более 4 посетителей, каждый из которых может сделать заказ тех или иных блюд. Столики обслуживают официанты. У одного официанта в обслуживании несколько столов.

Задачи для БД:

* Есть ли свободные столы?
* Сколько посетителей обслужил официант за смену?
  + Сколько каких блюд было реализовано? Вариант 2.

Описание предметной области (Колледж).

Студенты колледжа объединены в группы. Набор дисциплин, изучаемых студентом, зависит от номера группы в которой он учится. Преподаватели читают дисциплины и вы- ставляют зачеты студентам. Один преподаватель может читать несколько дисциплин, но каждую дисциплину ведет один преподаватель.

Задачи для БД:

* Какие дисциплины изучает студент?
* Какая оценка у студента по данной дисциплине?
  + Кто выставил эту оценку? Вариант3.

Описание предметной области (Театральная касса).

В театральной кассе продаются билеты на спектакли. Стоимость билета зависит от ряда, театра и спектакля. Каждый день в театре может идти не более одного спектакля. Спектакль характеризуется названием и автором. Каждый покупатель может купить сколько угодно билетов на любые спектакли.

Задачи для БД:

* Какие спектакли идут в определенный день?
* Есть ли билеты на конкретный спектакль?
  + Сколько стоит конкретный билет? Вариант 4.

Описание предметной области (Грузоперевозки).

АТП имеет грузовые автомобили с гос. номерами и организует перевозки для своих заказчиков. Стоимость перевозки зависит от расстояния и грузоподъемности автомобиля, который ее выполняет. Каждый заказчик может сделать заказ нескольких перевозок. Одну перевозку выполняет один грузовик.

Задачи для БД:

* Какие грузовики свободны?
* Какой заказчик сделал самый дорогой заказ?
  + Какой грузовик выполнил наибольшее количество заказов? Технология выполнения работы

1. Построение ER-диаграммы.
   1. Выберите из описания предметной области все существительные. Проду- майте, какие из них будут соответствовать сущностям, а какие атрибутам сущностей. Зари- суйте в отчет все сущности с их атрибутами согласно обозначениям, принятым в ER-диа- граммах.
   2. На рисунке подчеркиванием атрибутов обозначьте для каждой сущности уникальный идентификатор (Ключ). При необходимости добавьте сущностям атрибуты,

которые помогут однозначно отличить каждый экземпляр сущности.

* 1. Определите и включите в схему связи сущностей. Подпишите названия связей и пронумеруйте связи. Для первой связи укажите тип и модальность. Для всех связей запишите их прочтение слева направо и справа налево.
  2. Если в схеме присутствуют связи типа «много-со-многими» уберите их пу- тем ввода дополнительной сущности. Измененную схему зарисуйте в отчет.

1. Получение реляционной схемы из ER-диаграммы.
   1. Каждая сущность превращается в таблицу. Имя сущности – имя таблицы. Набор всех таблиц – БД. Вспомните, что такое схема БД. Запишите схему вашей БД в отчет.
   2. Зарисуйте все полученные таблицы с их заголовками и названиями столб- цов. Выделите потенциальные и внешние ключи (если есть) для каждой таблицы. Укажите столбцы, допускающие неопределенные значения.
   3. Докажите, что полученные отношения находятся в Первой нормальной форме.

## Практическая работа № 3 Нормализация реляционной БД, освоение принци-

**пов проектирования БД**

**Цель работы:** выработать практические навыки моделирования предметной обла- сти и построения различных видов модели баз данных Нормализация, функциональные и многозначные зависимости

Нормализация – это разбиение таблицы на две или более, обладающих лучшими свойствами при включении, изменении и удалении данных. Окончательная цель нормали- зации сводится к получению такого проекта базы данных, в котором каждый факт появля- ется лишь в одном месте, т.е. исключена избыточность информации. Это делается не столько с целью экономии памяти, сколько для исключения возможной противоречивости хранимых данных.

Каждая таблица в реляционной БД удовлетворяет условию, в соответствии с кото- рым в позиции на пересечении каждой строки и столбца таблицы всегда находится един- ственное атомарное значение, и никогда не может быть множества таких значений. Любая таблица, удовлетворяющая этому условию, называется нормализованной. Фактически, не- нормализованные таблицы, т.е. таблицы, содержащие повторяющиеся группы, даже не до- пускаются в реляционной БД.

Всякая нормализованная таблица автоматически считается таблицей в первой нор- мальной форме, сокращенно 1НФ. Таким образом, строго говоря, "нормализованная" и "находящаяся в 1НФ" означают одно и то же. Однако на практике термин "нормализован- ная" часто используется в более узком смысле – "полностью нормализованная", который означает, что в проекте не нарушаются никакие принципы нормализации.

Теперь в дополнение к 1НФ можно определить дальнейшие уровни нормализации – вторую нормальную форму (2НФ), третью нормальную форму (3НФ) и т.д.

По существу, таблица находится в 2НФ, если она находится в 1НФ и удовлетворяет, кроме того, некоторому дополнительному условию, суть которого будет рассмотрена ниже. Таблица находится в 3НФ, если она находится в 2НФ и, помимо этого, удовлетворяет еще другому дополнительному условию и т.д.

Таким образом, каждая нормальная форма является в некотором смысле более огра- ниченной, но и более желательной, чем предшествующая. Это связано с тем, что "(N+1)-я нормальная форма" не обладает некоторыми непривлекательными особенностями, свой- ственным "N-й нормальной форме". Общий смысл дополнительного условия, налагаемого на (N+1)-ю нормальную форму по отношению к N-й нормальной форме, состоит в исклю- чении этих непривлекательных особенностей.

За время развития технологии проектирования реляционных БД были выделены сле- дующие нормальные формы:

* первая нормальная форма (1NF);
* вторая нормальная форма (2NF);
* третья нормальная форма (3NF);
* нормальная форма Бойса-Кодда (BCNF);
* четвертая нормальная форма (4NF);
* пятая нормальная форма, или нормальная форма проекции-соединения (5NF).

Обычно на практике применение находят только первые три нормальные формы.

Теория нормализации основывается на наличии той или иной зависимости между полями таблицы. Определены два вида таких зависимостей: функциональные и многознач- ные.

Определение. Функциональная зависимость. Поле В таблицы функционально зави- сит от поля А той же таблицы в том и только в том случае, когда в любой заданный момент времени для каждого из различных значений поля А обязательно существует только одно из различных значений поля В. Отметим, что здесь допускается, что поля А и В могут быть составными.

Другими словами, в отношении R атрибут Y функционально зависит от атрибута X в том и только в том случае, если каждому значению X соответствует одно значение Y.

Схематично функциональную зависимость атрибута Y от атрибута X изображают

гак:

R.X -> R.Y

R(X -> Y).

ФЗ(X -> Y)

Пример 1.

* + 1. В таблице Блюда (б.д. Пансион) поля Блюдо и В функционально зависят от ключа БЛ.
    2. Таблица Поставщики вида:

Таблица 2.1. Поставщики

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Поставщик | Статус | Город | Страна | Адрес | Телефон |

В таблице Поставщики поле Страна функционально зависит от составного ключа (Поставщик, Город). Однако последняя зависимость не является функционально полной, так как Страна функционально зависит и от части ключа – поля Город.

Отсюда определение:

Определение. Полная функциональная зависимость. Поле В находится в полной функциональной зависимости от составного поля А, если оно функционально зависит от А и не зависит функционально от любого подмножества поля А.

Пример нормализации

Постановка задачи. Дано отношение.

1. определить первичный ключ отношения и все функциональные зависимости отношения;
2. привести отношение к 3НФ, указать первичные и внешние ключи полученных от- ношений, построить схему "Таблица-Связь".

R = {НаименованиеЭмитента, ТипЦБ, Да- таЭмиссии, НоминальнаяCтоимость}.

Таблица 2.2. Эмитенты

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование Эмитента | ТипЦБ | Дата Эмиссии | Номинальная Стоимость |
| ОАО ―КрАЗ‖ | акция обыкновенная | 23.06.1999 | 100 руб. |
| ОАО ―КрАЗ‖ | акция обыкновенная | 23.06.1999 | 200 руб. |
| ТОО ―Искра‖ | акция привилегирванная | 20.06.1999 | 500 руб. |
| ТОО ―Искра‖ | акция привилегированная | 23.06.1999 | 500 руб. |

Решение

1. Функциональные зависимости:

<ДатаЭмиссии, НоминальнаяCтоимость> -> <НаименованиеЭмитента, ТипЦБ> ТипЦБ -

>НаименованиеЭмитента

Первичный ключ отношения R состоит из двух атрибутов:

<ДатаЭмиссии, НоминальнаяCтоимость>.

Таким образом, существует функциональная зависимость между неключевыми ат- рибутами отношения R, т.е. отношение R не находится в 3НФ.

1. Приведение отношения R к 3НФ состоит в декомпозиции (разбиении отношения R на два отношения):

R1= {НаименованиеЭмитента, ТипЦБ}, где Функциональные зависимости: ТипЦБ -> НаименованиеЭмитента, первичный ключ – атрибут ТипЦБ,

R2={ТипЦБ, ДатаЭмиссии, НоминальнаяCтоимость}, Функциональные зависимости:

<ДатаЭмиссии, НоминальнаяCтоимость> -> ТипЦБ, составной первичный ключ:

<ДатаЭмиссии, НоминальнаяCтоимость>, внешний ключ:

ТипЦБ.

* + 1. Схема "Таблица-Связь":

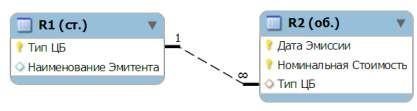


Рисунок 2.1. Схема «Таблица – связь»

### Задание для практической работы

Дан фрагмент отношения (таблицы). Предполагается, что функциональные зави- симости, имеющиеся во фрагменте, распространяются на все отношение (таблицу). Для ва- шего [варианта](http://orloff.am.tpu.ru/labs_itkd/kr2/index2.htm):

* + - 1. Определить первичный ключ отношения и все функциональные зависимости отношения.
      2. Привести отношение к 3НФ, указать первичные и внешние ключи полученных отношений, построить схему "Таблица - Связь".

### Индивидуальные задания к практической работе

Вариант 1.

|  |  |  |  |
| --- | --- | --- | --- |
| Область | Тип | C2N | Количество |
| А | С | 27 | 5 |
| А | С | 27 | 6 |
| Б | Д | 26 | 7 |
| Б | Д | 27 | 7 |

Вариант 2.

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование | Свойство | Артикул | Количество |
| А | С | 27 | 9 |
| А | Д | 27 | 9 |
| Б | С | 27 | 9 |
| Б | Д | 25 | 9 |

Вариант 3.

|  |  |  |  |
| --- | --- | --- | --- |
| Тип | Характеристика | Группа | Количество |
| А | С | 13 | 100 |
| А | Д | 13 | 100 |
| Б | С | 13 | 200 |
| Б | Д | 13 | 100 |

Вариант 4.

|  |  |  |
| --- | --- | --- |
| НаименованиеЭмитента | ТипЦБ | ДатаЭмиссии |
| ОАО ―КрАЗ‖ | акция привилегированная | 20.06.1999 |
| ОАО ―КрАЗ‖ | акция обыкновенная | 20.06.1999 |
| ЗАО ―Агат‖ | акция привилегированная | 23.06.1999 |
| ТОО ―Искра‖ | акция привилегированная | 20.06.1999 |

Вариант 5.

|  |  |  |
| --- | --- | --- |
| НаименованиеВуза | Команда | Приз |
| КрасГУ | МатФак | Торт |

|  |  |  |
| --- | --- | --- |
| КГТУ | ЭконФак | Арбуз |
| СибГТУ | МатФак | Торт |
| НГУ | ЭконФак | Бананы |

Вариант 6.

|  |  |  |
| --- | --- | --- |
| НаименованиеВуза | Адрес | Команда |
| КрасГУ | Красноярск, пр-т Свободный,79 | МатФак |
| КрасГУ | Красноярск, пр-т Свободный,79 | ЭконФак |
| КГТУ | Красноярск, ул.Киренского,26 | ЭконФак |
| КГТУ эк. институт | Красноярск, ул.Киренского,26 | ФизФак |
| НГУ | Новосибирск, ул. Пирогова, 2 | МатФак |
| НГУ | Новосибирск, ул. Пирогова, 2 | ФизФак |

Вариант 7.

|  |  |  |
| --- | --- | --- |
| НаименованиеВуза | Команда | Приз |
| КрасГУ | МатФак | Торт |
| КрасГУ | ЭконФак | Торт |
| НГУ | ФилФак | Арбуз |
| НГУ | МатФак | Бананы |

Вариант 8.

|  |  |  |
| --- | --- | --- |
| НаименованиеЭмитента | ТипЦБ | ДатаЭмиссии |
| ОАО ―КрАЗ‖ | акция привилегированная | 24.06.1999 |
| ЗАО ―Сибириада‖ | акция обыкновенная | 25.06.1999 |
| ТОО ―Искра‖ | акция привилегированная | 23.06.1999 |
| ЗАО ―Агат‖ | акция обыкновенная | 25.06.1999 |

Вариант 9.

|  |  |  |
| --- | --- | --- |
| Наименование Вуза | Адрес | Команда |
| КрасГУ | Красноярск, пр-т Свободный,79 | МатФак |
| КрасГУ | Красноярск, пр-т Свободный,79 | ЭконФак |
| КГТУ | Красноярск, ул.Киренского,26 | ФизФак |
| КГТУ эк. институт | Красноярск, ул.Киренского,26 | ЮрФак |
| НГУ | Новосибирск, ул. Пирогова, 2 | ФилФак |
| НГУ | Новосибирск, ул. Пирогова, 2 | БиоХим |

Вариант 10.

|  |  |  |
| --- | --- | --- |
| НаименованиеЭмитента | ТипЦБ | ДатаЭмиссии |
| ОАО ―КрАЗ‖ | акция привилегированная | 20.06.1999 |
| ОАО ―КрАЗ‖ | акция обыкновенная | 23.06.1999 |
| ЗАО ―Агат‖ | акция привилегированная | 20.06.1999 |
| ЗАО ―Агат‖ | акция привилегированная | 24.06.1999 |

Вариант 11.

|  |  |  |
| --- | --- | --- |
| P1 | P2 | Дата |
| A | D | 2000 |
| A | E | 2000 |
| B | D | 2005 |
| C | D | 2000 |

Вариант 12.

|  |  |  |
| --- | --- | --- |
| A | B | C |
| A1 | B1 | C1 |
| A2 | B2 | C2 |
| A3 | B1 | C1 |

|  |  |  |
| --- | --- | --- |
| A4 | B2 | C3 |

Вариант 13.

|  |  |  |
| --- | --- | --- |
| A | B | C |
| A1 | B1 | C1 |
| A1 | B1 | C2 |
| A2 | B2 | C2 |
| A3 | B2 | C3 |
| A4 | B3 | C1 |
| A4 | B3 | C3 |

Вариант 14.

|  |  |  |
| --- | --- | --- |
| A | B | C |
| A1 | B1 | C1 |
| A1 | B2 | C1 |
| A2 | B3 | C2 |
| A2 | B1 | C3 |

Вариант 15.

|  |  |  |
| --- | --- | --- |
| A | B | C |
| A1 | B1 | C1 |
| A2 | B2 | C2 |
| A3 | B1 | C3 |
| A4 | B2 | C2 |

Вариант 16.

|  |  |  |
| --- | --- | --- |
| A | B | C |
| A1 | B1 | C1 |
| A1 | B1 | C2 |
| A2 | B2 | C3 |
| A3 | B2 | C4 |
| A4 | B3 | C5 |
| A4 | B3 | C6 |

Вариант 17.

|  |  |  |
| --- | --- | --- |
| A | B | C |
| A1 | B1 | C1 |
| A1 | B2 | C2 |
| A2 | B1 | C1 |
| A2 | B1 | C3 |

Вариант 18.

|  |  |  |
| --- | --- | --- |
| P | F | Q |
| 41 | 21 | 11 |
| 42 | 22 | 12 |
| 43 | 21 | 11 |
| 44 | 22 | 13 |

Вариант 19.

|  |  |  |
| --- | --- | --- |
| C | D | E |
| 71 | 61 | 51 |
| 71 | 61 | 52 |
| 72 | 62 | 52 |
| 73 | 62 | 53 |
| 74 | 63 | 51 |

|  |  |  |
| --- | --- | --- |
| 74 | 63 | 53 |

Вариант 20.

|  |  |  |
| --- | --- | --- |
| F | G | H |
| App | Beep8 | Call7 |
| App | Beep2 | Call7 |
| Opp | Beep3 | Call2 |
| Opp | Beep8 | Call3 |

Вариант 21

|  |  |  |
| --- | --- | --- |
| НаименованиеВуза | Адрес | Команда |
| КрасГУ | Красноярск, пр-т Свободный, 79 | Первая |
| КрасГУ | Красноярск, пр-т Свободный, 79 | Вторая |
| КГТУ | Красноярск, ул. Киренского, 26 | Первая |
| КГТУ | Красноярск, ул. Киренского, 26 | Вторая |
| НГУ | Новосибирск, ул. Пирогова, 2 | Первая |
| НГУ | Новосибирск, ул. Пирогова, 2 | Вторая |

Вариант 22.

|  |  |  |  |
| --- | --- | --- | --- |
| H2O | Виртуальность | C2N | Масса |
| Yes | P | 27 | 45 |
| Yes | P | 27 | 91 |
| No | D | 26 | NULL |
| No | D | 27 | NULL |

Вариант 23.

|  |  |  |  |
| --- | --- | --- | --- |
| Количество | Свойство | Артикул | Наименование |
| 752 | NP | 2007 | HDR |
| 752 | PN | 2007 | HDR |
| 345 | NP | 2007 | HDR |
| 345 | PN | 2005 | HDR |

Вариант 24.

|  |  |  |  |
| --- | --- | --- | --- |
| Тип | Признак | Месяц | Век |
| Второй | Да | 10 | 19 |
| Второй | Нет | 10 | 19 |
| Первый | Да | 10 | 20 |
| Первый | Нет | 10 | 19 |

Вариант 25.

|  |  |  |
| --- | --- | --- |
| НаименованиеЭмитента | ТипЦБ | ДатаЭмиссии |
| ОАО ―Камаз‖ | акция привилегированная | 20.06.2010 |
| ОАО ―КрАЗ‖ | акция обыкновенная | 23.06.2010 |
| ЗАО ―Агат‖ | акция привилегированная | 20.06.2010 |
| ТОО ―Искра‖ | акция привилегированная | 20.06.2010 |

Вариант 26.

|  |  |  |
| --- | --- | --- |
| НаименованиеВуза | Команда | Приз |
| ТГУ | ФилФак | Сист. блок |
| ТПУ | ЭконФак | Ноутбук |
| ТГПУ | МатФак | Монитор |
| ТУСУР | ЭконФак | Ноутбук |

Вариант 27.

|  |  |  |
| --- | --- | --- |
| НаименованиеВуза | Адрес | Команда |
| ТГУ | Томск, пр. Ленина, 36 | МатФак |

|  |  |  |
| --- | --- | --- |
| ТГУ | Томск, пр. Ленина, 36 | ГРФак |
| ТПУ | Томск, пр. Ленина, 2 | ГРФак |
| ТПУ ФТИ | Томск, пр. Ленина, 2 | ФизФак |
| ТУСУР | Томск, пр. Ленина, 40 | ФизФак |
| ТУСУР | Томск, пр. Ленина, 40 | МатФак |

Вариант 28.

|  |  |  |
| --- | --- | --- |
| НаименованиеВуза | Команда | Приз |
| ТГУ | ЭконФак | Торт |
| ТГУ | МатФак | Торт |
| ТПУ | МатФак | Бананы |
| ТПУ | ФилФак | Арбуз |

Вариант 29.

|  |  |  |
| --- | --- | --- |
| НаименованиеЭмитента | ТипЦБ | ДатаЭмиссии |
| ЗАО ―Агат‖ | акция привилегированная | 13.05.2009 |
| ЗАО ―Сибириада‖ | акция обыкновенная | 14.05.2009 |
| ТОО ―Искра‖ | акция привилегированная | 12.05.2009 |
| ОАО ―КрАЗ‖ | акция обыкновенная | 14.05.2009 |

Вариант 30.

|  |  |  |
| --- | --- | --- |
| Организация | Адрес | Статус |
| ТРЕНД | Москва, Кирова, 79 | РОА |
| ТРЕНД | Москва, Кирова, 79 | ОУП |
| ЮРАС, первый | Томск, Кирова, 26 | ООО |
| ЮРАС | Томск, Кирова, 26 | ОАО |
| БЕППО | Юрга, Кирова, 2 | БПФ |
| БЕППО | Юрга, Кирова, 2 | ПРС |

Вариант 31.

|  |  |  |
| --- | --- | --- |
| Город | Событие | Дата |
| Калуга | пожар | 11.09.1812 |
| Калуга | наводнение | 14.09.1813 |
| Вятка | пожар | 11.09.1812 |
| Вятка | пожар | 15.09.1825 |

Вариант 32.

|  |  |  |
| --- | --- | --- |
| B1 | B2 | B3 |
| post | cool | stop |
| post | hot | stop |
| put | cool | stop |
| get | cool | nonstop |

Вариант 33.

|  |  |  |
| --- | --- | --- |
| A | B | C |
| 14 | 362 | 154 |
| 509 | 412 | 678 |
| 648 | 362 | 154 |
| 3 | 412 | 4 |

Вариант 34.

|  |  |  |
| --- | --- | --- |
| Риск | Фреш | Серия |
| Р1 | Ф1 | С2 |
| Р1 | Ф1 | С1 |
| Р2 | Ф2 | С2 |

|  |  |  |
| --- | --- | --- |
| Р3 | Ф2 | С3 |
| Р4 | Ф3 | С3 |
| Р4 | Ф3 | С1 |

Вариант 35.

|  |  |  |
| --- | --- | --- |
| Тип | Вес | Цена |
| HDR | 512 | 13500 |
| HDR | 432 | 13500 |
| GTS | 256 | 14000 |
| GTS | 512 | 16000 |

Вариант 36.

|  |  |  |
| --- | --- | --- |
| A | B | C |
| A11 | B7 | C2 |
| A21 | B7 | C1 |
| A17 | B9 | C4 |
| A3 | B9 | C4 |

Вариант 37.

|  |  |  |
| --- | --- | --- |
| A | B | C |
| 111 | 31 | 82 |
| 101 | 31 | 3 |
| 45 | 77 | 2 |
| 45 | 77 | 7 |
| 92 | 91 | 54 |
| 92 | 91 | 115 |

Вариант 38.

|  |  |  |
| --- | --- | --- |
| Red | Blue | Green |
| F101 | 567 | 706 |
| F101 | A812 | 535 |
| C255 | 567 | 706 |
| C255 | 567 | 536 |

Вариант 39.

|  |  |  |
| --- | --- | --- |
| D | F | T |
| 284 | 17 | 78 |
| 968 | 17 | 89 |
| 274 | 33 | 56 |
| 904 | 33 | 56 |

Вариант 40.

|  |  |  |
| --- | --- | --- |
| AA | YY | ZZ |
| 72 | 62 | 52 |
| 71 | 61 | 52 |
| 73 | 62 | 53 |
| 74 | 63 | 53 |
| 74 | 63 | 51 |
| 71 | 61 | 51 |

Вариант 41.

|  |  |  |  |
| --- | --- | --- | --- |
| Наличие | Тип | Число | Масса |
| Yes | FULL | 45 | 90 |
| No | DEPT | 54 | 130 |
| No | DEPT | 45 | 130 |

|  |  |  |  |
| --- | --- | --- | --- |
| Yes | FULL | 45 | 60 |

Вариант 42.

|  |  |  |  |
| --- | --- | --- | --- |
| Вероятность\*1000 | Время, с | Мощность, вт | Тип |
| 694 | 15 | 2015 | HDR |
| 694 | 17 | 2015 | HDR |
| 604 | 15 | 3500 | HPQ |
| 604 | 17 | 2015 | HDR |

Вариант 43.

|  |  |  |
| --- | --- | --- |
| Тип | Наличие | Артикул |
| Второй | Да | 17 |
| Второй | Нет | 17 |
| Первый | Да | 17 |
| Первый | Нет | 17 |
| Второй | Неизвестно | 18 |
| Первый | Неизвестно | 18 |

Вариант 44.

|  |  |  |
| --- | --- | --- |
| A | B | C |
| 17 | 49 | 17 |
| 35 | 53 | 35 |
| 24 | 49 | 24 |
| 19 | 53 | 35 |

ласти,

**Практическая работа №4** [**Проектирование реляционной БД. Нормализация таб-**](#_bookmark1)[**лиц**](#_bookmark1)**. Задание ключей. Создание основных объектов БД»**

**Цель работы:** приобретение практических навыков анализа предметной об-

информационных задач и построения концептуальной модели базы данных.

### Проектирование базы данных (БД)

Проектирование базы данных (БД) – одна из наиболее сложных и ответственных за-

дач, связанных с созданием информационной системы (ИС). В результате еѐ решения должны быть определены содержание БД, эффективный для всех еѐ будущих пользовате- лей способ организации данных и инструментальные средства управления данными.

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

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

Внимание! Базы данных всегда проектируются под конкретное назначение системы. Техника проектирования баз данных может измениться в целом и в деталях в зави- симости от назначения системы. Например, следует различать проектирование систем скла- дирования данных и проектирование так называемых OLTP-систем, ориентируемых на опе- ративную обработку транзакций. В данном учебном курсе рассматривается проектирование баз данных в основном для OLTP-систем. Именно на таких системах исторически сложи-

лась техника проектирования баз данных.

### Этапы проектирования базы данных

Процесс проектирования включает в себя следующие этапы:

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

Логическое проектирование – это процесс конструирования информационной мо- дели на основе существующих моделей данных, не зависимо от используемой СУБД и дру- гих условий физической реализации.

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

Концептуальное проектирование

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

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

Рассмотрим основные подходы к созданию концептуальной модели предметной об-

ласти.

1. Функциональный подход к проектированию БДЭтот метод реализует принцип

"от задач" и применяется тогда, когда известны функции некоторой группы лиц и/или ком- плекса задач, для обслуживания информационных потребностей которых создаѐтся рас- сматриваемая БД.

1. Предметный подход к проектированию БД

Предметный подход к проектированию БД применяется в тех случаях, когда у раз- работчиков есть чѐткое представление о самой ПО и о том, какую именно информацию они хотели бы хранить в БД, а структура запросов не определена или определена не полностью. Тогда основное внимание уделяется исследованию ПО и наиболее адекватному еѐ отобра- жению в БД с учѐтом самого широкого спектра информационных запросов к ней.

1. Проектирование с использованием метода "сущность-связь"

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

ется на локальные области таким образом, чтобы каждая из них соответствовала отдель- ному внешнему приложению и содержала 6-7 сущностей.

Сущность – это объект, о котором в системе будет накапливаться информация. Сущ- ности бывают как физически существующие (например, СОТРУДНИК или АВТОМО- БИЛЬ), так и абстрактные (например, ЭКЗАМЕН или ДИАГНОЗ). Для сущностей разли- чают тип сущности и экземпляр. Тип характеризуется именем и списком свойств, а экзем- пляр – конкретными значениями свойств. Типы сущностей можно классифицировать как сильные и слабые. Сильные сущности существуют сами по себе, а существование слабых сущностей зависит от существования сильных. Например, читатель библиотеки – сильная сущность, а абонемент этого читателя – слабая, которая зависит от наличия соответствую- щего читателя. Слабые сущности называют подчинѐнными (дочерними), а сильные – базо- выми (основными, родительскими).

Для каждой сущности выбираются свойства (атрибуты). Различают:

* + Идентифицирующие и описательные атрибуты. Идентифицирующие атрибуты имеют уникальное значение для сущностей данного типа и являются потенциальными ключами. Они поз- воляют однозначно распознавать экземпляры сущности. Из потенциальных ключей выбирается один первичный ключ (ПК). В качестве ПК обычно выбирается потенциальный ключ, по которому чаще происходит обращение к экземплярам записи. Кроме того, ПК должен включать в свой состав минимально необходимое для идентификации количество атрибутов. Остальные атрибуты называ- ются описательными и заключают в себе интересующие свойства сущности.
  + Составные и простые атрибуты. Простой атрибут состоит из одного компонента, его значение неделимо. Составной атрибут является комбинацией нескольких компонентов, возможно, принадлежащих разным типам данных (например, ФИО или адрес). Решение о том, использовать составной атрибут или разбивать его на компоненты, зависит от характера его обработки и формата пользовательского представления этого атрибута.
  + Однозначные и многозначные атрибуты (могут иметь соответственно одно или много значений для каждого экземпляра сущности).
  + Основные и производные атрибуты. Значение основного атрибута не зависит от дру- гих атрибутов. Значение производного атрибута вычисляется на основе значений

других атрибутов (например, возраст студента вычисляется на основе даты его рож- дения и текущей даты).

Спецификация атрибута состоит из его названия, указания типа данных и описания ограничений целостности – множества значений (или домена), которые может принимать данный атрибут. Далее осуществляется спецификация связей внутри локального представ- ления. Связи могут иметь различный содержательный смысл (семантику). Различают связи типа "сущность- сущность", "сущность-атрибут" и "атрибут-атрибут" для отношений между атрибутами, которые характеризуют одну и ту же сущность или одну и ту же связь

типа "сущность-сущность". Каждая связь характеризуется именем, обязательностью, типом и степенью. Различают факультативные и обязательные связи. Если вновь порождѐнный объект одного типа оказывается по необходимости связанным с объектом другого типа, то между этими типами объектов существует обязательная связь (обозначается двойной ли- нией). Иначе связь является факультативной. По типу различают множественные связи "один к одному" (1:1), "один ко многим" (1:N) и "многие ко многим" (M:N). Степень связи определяется количеством сущностей, которые охвачены данной связью. Пример бинарной связи – связь между отделом и сотрудниками, которые в нѐм работают. Примером тернар- ной связи является связь типа экзамен между сущностями ДИСЦИПЛИНА, СТУДЕНТ, ПРЕПОДАВАТЕЛЬ. Из последнего примера видно, что связь также может иметь атрибуты (в данном случае это Дата проведения и Оценка). Пример ER–диаграммы с указанием сущ- ностей, их атрибутов и связей приведен на рис. 1.



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

# Пример проектирования реляционной базы данных

В качестве примера возьмем базу данных компании, которая занимается издатель- ской деятельностью. База данных создаѐтся для информационного обслуживания редакто- ров, менеджеров и других сотрудников компании. БД должна содержать данные о сотруд- никах компании, книгах, авторах, финансовом состоянии компании и предоставлять воз- можность получать разнообразные отчѐты. В соответствии с предметной областью система строится с учѐтом следующих особенностей:

* + каждая книга издаѐтся в рамках контракта;
  + книга может быть написана несколькими авторами;
  + контракт подписывается одним менеджером и всеми авторами книги;
  + каждый автор может написать несколько книг (по разным контрактам);
  + порядок, в котором авторы указаны на обложке, влияет на размер гонорара;
  + если сотрудник является редактором, то он может работать одновременно над несколькими книгами;
  + у каждой книги может быть несколько редакторов, один из них – ответственный редактор;
  + каждый заказ оформляется на одного заказчика;в заказе на пкупку может быть пере- числено несколько книг. Выделим базовые сущности этой предметной области:

1. Сотрудники компании. Атрибуты сотрудников – ФИО, табельный номер, пол, дата рождения, паспортные данные, ИНН, должность, оклад, домашний адрес и телефоны. Для редакто- ров необходимо хранить сведения о редактируемых книгах; для менеджеров – сведения о подпи- санных контрактах.
2. Авторы. Атрибуты авторов – ФИО, ИНН (индивидуальный номер налогоплатель- щика), паспортные данные, домашний адрес, телефоны. Для авторов необходимо хранить сведения о написанных книгах.
3. Книги. Атрибуты книги – авторы, название, тираж, дата выхода, цена одного экзем- пляра, общие затраты на издание, авторский гонорар.
4. Контракты будем рассматривать как связь между авторами, книгами и менеджерами.

Атрибуты контракта – номер, дата подписания и участники.

1. Для отражения финансового положения компании в системе нужно учитывать заказы на книги. Для заказа необходимо хранить номер заказа, заказчика, адрес заказчика, дату поступле- ния заказа, дату его выполнения, список заказанных книг с указанием количества экземпляров.

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

Анализ информационных задач и круга пользователей системы Система создаѐтся для обслуживания следующих групп пользователей:

* + администрация (дирекция);
  + менеджеры;
  + редакторы;
  + сотрудники компании, обслуживающие заказы. Определим границы информационной поддержки пользователей:

1. Функциональные возможности:
   * ведение БД (запись, чтение, модификация, удаление в архив);
   * обеспечение логической непротиворечивости БД;
   * обеспечение защиты данных от несанкционированного или случайного доступа (определение прав доступа);
   * реализация наиболее часто встречающихся запросов в готовом виде;
   * предоставление возможности сформировать произвольный запрос на языке манипулирования данными.
2. Готовые запросы:
   * получение списка всех текущих проектов (книг, находящихся в печати и в продаже);
   * получение списка редакторов, работающих над книгами;
   * получение полной информации о книге (проекте);
   * получение сведений о конкретном авторе (с перечнем всех книг);
   * получение информации о продажах (по одному или по всем проектам);
   * определение общей прибыли от продаж по текущим проектам;
   * определение размера гонорара автора по конкретному проекту.

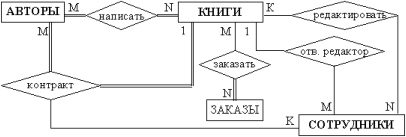


Рисунок 2. ER–диаграмма издательской компании

***Задание для практической работы***

## По заданному описанию предметной области построить концептуальную мо- дель базы данных

Выделите типы сущностей;

* + Выделите типы связей и определите для них показатели кардинальности и степень участия сторон;
  + Выделите атрибуты и свяжите их типами сущностей и связей;
  + Определите потенциальные и первичные ключи сущностей;
  + Нарисуйте ER-диаграмму.
  + и проанализируйте информационные задачи и группы пользователей.

### Индивидуальные задания к практической работе

Вариант 1.

Задача – организация учебного процесса в вузе:

* + Студенты: паспортные данные, адрес, дата зачисления, номер приказа, фа- культет, группа, является ли старостой, кафедра (специализация), изучаемые (изученные) предметы, оценки, задолженности, стипендия.
  + Учебные курсы: название, факультет(ы), групп(ы), кафедра, семестр(ы), форма отчѐтности, число часов.
  + Преподаватели: паспортные данные, адрес, телефон, фотография, кафедра, должность, учѐная степень, начальник (зав. кафедрой), предмет(ы), число ставок, зарплата.

Вариант 2.

Учет и выдача книг в библиотеке вуза:

* + Книги: авторы, название, раздел УДК, раздел (техническая, общественно- политическая и т.п.), место и год издания, издательство, количество страниц, иллюстриро- ванность, цена, дата покупки, номер сопроводительного документа (чек, счѐт/накладная), вид издания (книги, учебники, брошюры, периодические издания), инвентарный номер (есть только для книг и некоторых учебников), длительность использования читателями (год, две недели, день), электронная версия книги или ее реферата (отсканированный текст).
  + Читатели: номер читательского билета, ФИО, год рождения, адрес, дата за- писи, вид (студент, аспирант, преподаватель, сотрудник), курс, номер группы, названия взя- тых книг и даты их выдачи.
  + Вариант3.
  + Отдел кадров некоторой компании.
  + Сотрудники: ФИО, паспортные данные, фотография, дом. и моб. телефоны, отдел, комната, раб. телефоны (в т.ч. местный), подчинѐнные сотрудники, должность, тип(ы)работы, задание(я), проект(ы), размер зарплаты, форма зарплаты (почасовая, фикси- рованная).
  + Отделы: название, комната, телефон(ы), начальник, размер финансирова- ния, число сотрудников.
  + Проекты: название, дата начала, дата окончания, размер финансирования, тип финансирования (периодический, разовый), задачи и их исполнители, структура затрат и статьи расходов.
  + Вариант 4.
  + Отдел поставок некоторого предприятия.
  + Поставщики: название компании, ФИО контактного лица, расчѐтный счѐт в банке, телефон, факс, поставляемое оборудование (материалы), даты поставок (по догово- рам и реальные), метод и стоимость доставки.
  + Сырьѐ: тип, марка, минимальный запас на складе, время задержки, цена, продукты, при производстве которых используется, потребляемые объемы (необходимый, реальный, на единицу продукции).
  + Вариант5.
  + Пункт проката видеозаписей (внутренний учет).
  + Видеокассеты: идентификационный номер видеокассеты, тип видеокассет, дата его создания, компания-поставщик, число штук данного типа (общее, в магазине, вы- дано в настоящее время, выдано всего, выдано в среднем за месяц), общая длительность записей; записи видеокассет: название, длительность, категория, год выпуска и производи- тель (оригинала).
  + Клиенты: ФИО, паспортные данные, адрес, телефон; заказы, т.е. взятые ви- деокассеты (сейчас и в прошлом): номер, дата выдачи, дата возвращения, общая стоимость заказа.
  + Вариант 6.
  + Пункт проката видеозаписей (информация для клиентов).
  + Видеокассеты: краткое описание, внешний вид (этикетка), марка (пустой) видеокассеты, цена за единицу прокатного времени (например: 1 день, 3 дня, неделя), есть ли в наличии, общая длительность записей; записи на видеокассете: название, длитель- ность, жанр (категория), тема, год и страна выпуска (оригинала), кинокомпания, описание, актеры, режиссер.
  + Заказы: идентификационные номера и названия выданных видеокассет, дата выдачи, дата возвращения (продления), общая стоимость заказа, возвращены ли кас- сеты заказа.
  + Вариант 7.
  + Кинотеатры (информация для зрителей).
  + Фильмы: название, описание, жанр (категория), длительность, популяр- ность (рейтинг, число проданных билетов в России и в мире), показывается ли сейчас (се- годня, на текущей неделе), в каких кинотеатрах показывается, цены на билеты (в т.ч. сред- ние).
  + Кинотеатры: название, адрес, схема проезда, описание, число мест (в разных залах, если их несколько), акустическая система, широкоэкранность, фильмы и цены на них: детские и взрослые билеты в зависимости от сеанса (дневной, вечерний и т.п.) и от категории мест (передние, задние и т.п.); сеансы показа фильмов (дата и время начала).
  + Вариант 8.
  + Ресторан (информация для посетителей).
  + Меню: дневное или вечернее, список блюд по категориям.
* Блюда: цена, название, вид кухни, категории (первое, второе и т.п.; мясное, рыбное, салат и т.п.), является ли вегетарианским, компоненты блюда, время приготовления, есть ли в наличии. Компоненты блюд: тип (гарнир, соус, мясо и т.п.), калорийность, цена, рецепт, время приготовления, есть ли в наличии, ингредиенты (продукты) и их расходы на порцию.
  + Вариант 9.
  + Задача - информационная поддержка деятельности склада.
  + База данных должна содержать информацию о наименовании товара, его поставщике, количестве, цене товара, конечном сроке реализации, сроке хранения на складе. Торговый склад производит уценку хранящейся продукции. Если продукция хра- нится на складе дольше 10 месяцев, то она уценивается в 2 раза, а если срок хранения пре- высил 6 месяцев, но не достиг 10, то в 1,5 раза. Ведомость уценки товаров должна содер- жать информацию: наименование товара, количество товара(шт.), цена товара до уценки, срок хранения товара, цена товара после уценки, общая стоимость товаров после уценки.
  + Вариант10.
  + Задача – информационная поддержка деятельности адвокатской конторы. БД должна осуществлять:
  + ведение списка адвокатов;
  + ведение списка клиентов;
  + ведение архива законченных дел.
  + Необходимо предусмотреть:
  + получение списка текущих клиентов для конкретного адвоката;
  + определение эффективности защиты (максимальный срок минус полученный срок) с учѐтом оправданий, условных сроков и штрафов;
  + определение неэффективности защиты (полученный срок минус минималь- ный срок);
  + подсчѐт суммы гонораров (по отдельных делам) в текущем году;
  + получение для конкретного адвоката списка текущих клиентов, которых он защищал ранее (из архива, с указанием полученных сроков и статей).
  + Вариант11.
  + Задача – информационная поддержка деятельности гостиницы. БД должна осуществлять:
  + ведение списка постояльцев;
  + учѐт забронированных мест;
  + ведение архива выбывших постояльцев за последний год.
  + Необходимо предусмотреть:
  + получение списка свободных номеров (по количеству мест и классу);
  + получение списка номеров (мест), освобождающихся сегодня и завтра;
  + выдачу информации по конкретному номеру;
  + автоматизацию выдачи счетов на оплату номера и услуг;
  + получение списка забронированных номеров;
  + проверку наличия брони по имени клиента и/или названию организации Ва- риант12.
  + Описание предметной области:
  + В компании несколько отделов.
  + В каждом отделе есть некоторое количество сотрудников, занятых в не- скольких проектах и размещающихся в нескольких офисах.
  + Каждый сотрудник имеет план работы, т.е. несколько заданий, которые он должен выполнить. Для каждого такого задания существует ведомость, содержащая пере- чень денежных сумм, полученных сотрудником за выполнение этого задания.
  + В каждом офисе установлено несколько телефонов.
  + В базе данных должна храниться следующая информация.
  + Для каждого отдела: номер отдела ( уникальный), его бюджет и личный но- мер сотрудника, возглавляющего отдел ( уникальный).
  + Для каждого сотрудника: личный номер сотрудника ( уникальный), номер текущего проекта, номер офиса, номер телефона, название выполняемого задания вместе с датой и размером выплат, проведенных в качестве оплаты за выполнение данного задания.
  + Для каждого проекта: номер проекта ( уникальный) и его бюджет.
  + Для каждого офиса: номер офиса ( уникальный), площадь в квадратных фу- тах, номера всех установленных в нем телефонов.
  + Вариант13.
  + Задача – информационная поддержка деятельности спортивного клуба. БД должна осуществлять:
  + ведение списков спортсменов и тренеров;
  + учѐт проводимых соревнований (с ведением их архива);
  + учѐт травм, полученных спортсменами. Необходимо предусмотреть:
  + возможность перехода спортсмена от одного тренера к другому;
  + составление рейтингов спортсменов;
  + составление рейтингов тренеров;
  + выдачу информации по соревнованиям;
  + выдачу информации по конкретному спортсмену;
  + подбор возможных кандидатур на участие в соревнованиях (соответствую- щего уровня мастерства, возраста и без травм).
  + Вариант14.
  + Задача – информационная поддержка деятельности аптечного склада.
  + В аптечном складе хранятся лекарства. Сведения о лекарствах содержатся в специальной ведомости: наименование лекарственного препарата; количество (в шт.); цена; срок хранения на складе (в месяцах). Лекарства поступают на склад ежедневно от разных поставщиков, отпускаются два раза в неделю по предварительным заказам аптек. Выяснить, сколько стоит самый дорогой и самый дешевый препарат; сколько препаратов хранится на складе более 3 месяцев; сколько стоят все препараты, хранящиеся на складе, отыскать пре- параты, остаток которых равен нулю , ниже требуемого по заказам.
  + Вариант15.
  + ―Электронный журнал посещаемости"
  + Предметная область представлена следующими документами:
  + Список студентов
  + Журнал посещаемости
  + Расписание занятий
  + Предусмотреть учет пропусков по уважительным, неуважительным причи- нам. Подсчет
  + пропусков по каждому студенту, за неделю, месяц, заданный период, по кон- кретному
  + предмету. Вариант16.
  + «Итоги сессии»
  + База данных должна содержать информацию о двух последних сессиях сту- дентов. Источником информации являются экзаменационные ведомости. Необходимо про- водить анализ успеваемости по специальностям, формам обучения, курсам, группам, пред- метам, вычислять средний балл по указанным критериям, а также число каждых оценок .

## Практическая работа №5 Создание проекта БД. Создание БД. Редактирование и

**модификация таблиц**

**Цель работы:** освоить основные приемы заполнения и редактирования таблиц; по- знакомиться с простой сортировкой данных и с поиском записей по образцу; научиться со- хранять и загружать базы данных.

### Microsoft Office Access

Access является наиболее сложной программой из всех офисных приложений Microsoft Office. Чтобы начать работу с этой программой, вначале необходимо создать структуру базы данных, подробно ее описать, а затем создать различные формы.

ACCESS – это реляционная СУБД. Это означает, что с ее помощью можно работать одновременно с несколькими таблицами базы данных, эти таблицы между собой связаны. Таблицу ACCESS можно связать с данными, хранящимися на другом компьютере. Данные ACCESS очень просто комбинировать с данными EXCEL, WORD и другими программами Office.

Access во многом похож на Excel. Основное различие между таблицей БД и элек- тронной таблицей – в системе адресации: в электронной таблице адресуется каждая ячейка, а в таблице БД – только поля текущей записи.

Таблицы – основные объекты базы данных (БД). В них хранятся данные. Реляцион- ная база данных может иметь много взаимосвязанных таблиц. Сведения по разным вопро- сам следует хранить в разных таблицах.

Запрос – это средство, с помощью которого извлекается из базы данных информа- ция, отвечающая определенным критериям. Результаты запроса представляют не все записи из таблицы, а только те, которые удовлетворяют запросу.

Формы – Обеспечивают более наглядную работу с таблицами, с помощью форм в базу вводят новые данные или просматривают имеющиеся.

Отчеты – средство представления данных таблиц. Отчеты могут быть оформлены надлежащим образом и распечатаны в том виде, в котором требуется пользователю.

Макросы – набор из одной или более макрокоманд, выполняющих определенные операции (открытие форм, печать отчетов)

Модули *-* это программы, написанные на языке программирования Visual Basic.

### Задание для практической работы Заполните таблицы по образцу

1. Вызвать программу Access 2007 (Access 2016).
2. В окне системы управления базы данных щелкнуть по значку «Новая база данны**х**». Справа в появившемся окне дать имя новой базе данных «Анкета ГС-31» и щелк- нуть по значку папки, находящемуся справа от окна названия. Откроется окно сохране- ния, найдите свою папку и сохраните в нее новый файл базы данных «Анкета ГС-31» (вме- сто ГС-31 укажите номер вашей группы). Затем нажмите на кнопку

«Создать».

1. Появится окно «Таблица» (Рисунок 5.1).

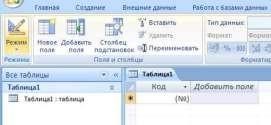


Рисунок 5.1. Окно пустой базы данных

1. В появившемся окне откройте меню команды <Режим> и выберите вариант



<Конструктор> и сохраните будущую таблицу под названием <Ве- домость успеваемости>. Появится окно Конструктора.

1. Заполните поля в Конструкторе данными из *рисунка 5.2.* Тип данных можно выбрать из



меню, появившемся при нажатии на кнопку в ячейке справа.

Обратите внимание*: ключевое поле «Счетчик» внесен в таблицу автоматически. Если напротив поля отсутствует значок ключа, то на панели инструментов щелкните по этому*

*значку. *

**

Рисунок 5.2. Создание таблицы через конструктор

1. Перейдите в режим таблицы, щелкнув по кнопке **Режим** на панели инстру- ментов, Введите данные в этом режиме, заполняя клетки таблицы. Значение поля **Код** будет меняться автоматически.
2. Заполните базу данных значениями из *таблицы 5.1*. Напротив каждой фа- милии выставьте по всем дисциплинам оценки от 2 до 5

Таблица 5.1.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Код | Фамилия | Имя | Матем атика | Менед жмент | Серви сная деятел ьность | Инфор мацио нные технол огии | Станд артиза ция | Гости нична я индуст рия | Пропу ски по неува жител ьной причи  не | Пропу ски по уважи тельно й причи не |
| 1 | Иваникова | Анна |  |  |  |  |  |  |  |  |
| 2 | Баранова | Ирина |  |  |  |  |  |  |  |  |
| 3 | Корнилова | Ольга |  |  |  |  |  |  |  |  |
| 4 | Воробьев | Алексей |  |  |  |  |  |  |  |  |
| 5 | Воробьев | Олег |  |  |  |  |  |  |  |  |
| 6 | Скоркин | Алекс |  |  |  |  |  |  |  |  |
| 7 | Володина | Нина |  |  |  |  |  |  |  |  |
| 8 | Новоселов | Алексей |  |  |  |  |  |  |  |  |
| 9 | Петрова | Елена |  |  |  |  |  |  |  |  |
| 10 | Чернова | Кристина |  |  |  |  |  |  |  |  |
| 11 | Терещинка | Инна |  |  |  |  |  |  |  |  |
| 12 | Истратов | Максим |  |  |  |  |  |  |  |  |
| 13 | Бондарь | Ольга |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 14 | Ревин | Олег |  |  |  |  |  |  |  |  |
| 15 | Шарова | Оксана |  |  |  |  |  |  |  |  |

1. Выполните редактирование ячеек:

– Замените фамилию Иванникова на Иванова.

1. Отсортируйте:

а) *фамилии* – по алфавиту (поставьте маркер на любую фамилию в столбце Фамилия и щелкнете мышкой по кнопке на панели инструментов или произ-



ведите сортировку с помощью контекстного меню) б) *имя* – по алфавиту

1. Сохраните текущую таблицу, щелкнув по кнопке «крестик» в правом верхнем углу окна таблицы.
2. Откройте снова свою базу данных.
3. Выполните поиск записей по образцу: *найти студентку по фамилии Во- лодина*. Для



этого установите курсор в поле фамилия, щелкните на кнопке **<**Би- нокль> на панели инструментов меню Главная и в появившемся диалоговом окне введите в поле

<Образец> фамилию *Володина* и щелкните по кнопке <Найти>.

Примечание: Если требуется найти следующую подобную запись, то щелкните мышкой по кнопке <Найти далее>. По окончании работы щелкните по кнопке <Отмена>.

1. Переименуйте поле «Математика» на «Информатика» с помощью кон- текстного меню. (Верните все как было назад).
2. Скройте столбец Пр н/пр., потом отобразите его назад.
3. Войдите в режим *Конструктора* и назначьте полю Пр н/пр и Пр ув/пр.

*Маску ввода* 00

«часов». Заполните эти поля данными от 0 до 99.

1. Завершите работу с Access.

## Практическая работа №6 Создание ключевых полей. Задание индексов.

**Установление и удаление связей между таблицами**

**Цели работы:** научиться самостоятельно создавать ключевое поле; закрепить навыки по удалению, добавлению, заполнению и редактированию таблиц; научиться ис- пользовать фильтр в таблице.

### Microsoft Office Access

Access является наиболее сложной программой из всех офисных приложений Microsoft Office. Чтобы начать работу с этой программой, вначале необходимо создать структуру базы данных, подробно ее описать, а затем создать различные формы.

ACCESS – это реляционная СУБД. Это означает, что с ее помощью можно работать одновременно с несколькими таблицами базы данных, эти таблицы между собой связаны. Таблицу ACCESS можно связать с данными, хранящимися на другом компьютере. Данные ACCESS очень просто комбинировать с данными EXCEL, WORD и другими программами Office.

Access во многом похож на Excel. Основное различие между таблицей БД и элек- тронной таблицей – в системе адресации: в электронной таблице адресуется каждая ячейка, а в таблице БД – только поля текущей записи.

Таблицы – основные объекты базы данных (БД). В них хранятся данные. Реляцион- ная база данных может иметь много взаимосвязанных таблиц. Сведения по разным вопро- сам следует хранить в разных таблицах.

Запрос – это средство, с помощью которого извлекается из базы данных информа- ция, отвечающая определенным критериям. Результаты запроса представляют не все записи из таблицы, а только те, которые удовлетворяют запросу.

Формы – Обеспечивают более наглядную работу с таблицами, с помощью форм в базу вводят новые данные или просматривают имеющиеся.

Отчеты – средство представления данных таблиц. Отчеты могут быть оформлены надлежащим образом и распечатаны в том виде, в котором требуется пользователю.

Макросы – набор из одной или более макрокоманд, выполняющих определенные операции (открытие форм, печать отчетов)

Модули *-* это программы, написанные на языке программирования Visual Basic.

***Задание для практической работы***

## Создайте таблицы и схему данных по заданным критериям

1. Откройте учебную базу данных <Анкета ГС-31>.
2. Создайте таблицу <Преподаватели > в *Режиме таблицы*. Для этого в меню Создание выберите кнопку Таблица. В появившейся таблице сделайте следующее:
   * Добавьте два поля – Поле 1 и Поле 2, выполнив команду через кон- текстное меню.
   * Переименуйте <Поле 1> на <Предмет>. Для этого поставьте курсор в любую ячейку столбца <Поля 1> и выполните команду *Переименовать столбец* из кон- текстного меню. Или щелкните два раза по имени поля, удалите старое название и впеча- тайте новое.
   * Переименуйте аналогично <Поле 2> на <Преподаватель>.
3. Сохраните таблицу с именем <Преподаватели>, щелкнув по кнопке <Сохра- нить> (дискетка на панели инструментов).
4. Перейдите в режим <Конструктор> и удалите строку с ключевым словом

Счетчик. Посмотрите как заданы поля. Сделайте поле <Предмет> ключевым, поместив кур- сор на имя этого поля и щелкнув по кнопке - *Ключевое поле*. Тип данных поля задайте *текстовым*.



1. Перейдите в *Режим таблицы* и заполните таблицу <Преподаватели> записями из

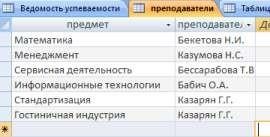
*Рисунок 6.1.*

Рисунок 6.1. Таблица «Преподаватели»

1. Закройте таблицу <Преподаватели>, сохранив все изменения.
2. Используя <Шаблон таблиц>, создайте таблицу <Личные данные> студен- тов с ключевым полем. Для этого:
   * Находясь на закладке <Создание> щелкните по кнопке <Шаблоны таблиц>,

<Контакты>. Появится таблица уже с готовыми полями.

* + Переименуйте предложенные поля на следующие поля: <Код студента>,

<Фамилия>, <Имя>, <Город>, <Адрес>, <Телефон>, <Дата рождения>,

<Фотография>, <Любимый предмет>, лишние поля удалите.

* + Сохраните полученную таблицу под названием <Личные данные>. Клю- чевое поле задано автоматически.

1. Внесите данные в новую таблицу, заполнив поля <Фамилия>, <Имя>,

<Город>,

<Адрес>, <Телефон>, <Дата рождения>.

ПРИМЕЧАНИЕ. Поля <Фамилия> и <Имя> можно скопировать из таблицы <Ведо- мость успеваемости>. В поле <Город> внесите четыре разных города (например, Новорос- сийск, Геленджик, Анапа, Крымск)

1. Перейдите в режим <Конструктор> и назначьте типы данных: для поля <Те- лефон> - *числовой*, для поля <Дата рождения> - *дата/время*, для поля <Фотография> – *поле объекта OLE, для остальных – текстовый.*

Для поля <Любимый предмет> выполните свойство выбор предмета из списка с по- мощью *Мастера подстановок*. Для этого в строке <Любимый предмет> в поле *Тип данных – текстовый* щелкните по кнопке и в ниспадающем меню выберите команду <Мастер подстановок>.



* + В диалоговом окне <Создание подстановки> поставьте флажок напротив способа

<*Будет введен фиксированный набор значений*> и нажмите <Далее>.

* + В следующем окне внесите в столбец все предметы (предметы из таблицы

<Преподаватели>), нажмите <Далее>.

* + В последнем окне, не изменяя имени столбца нажмите <Готово>.

1. Перейдите в режим таблицы и выберите для каждого студента с помощью кнопки из списка любимый предмет.



1. Создайте *схему данных*, т.е. установите связи между таблицами.
   * Щелкните по кнопке - *Схема данных* на панели инструментов меню



<Работа с базами данных>. В окне <Отобразить таблицу> выделите таблицу <Ведомость успеваемости> и щелкните по кнопке <Добавить>. Также добавьте таблицы

<Преподаватели> и <Личные данные>. В окне <Схема данных> появиться условный вид этих таблиц. Закройте окно <Добавление таблицы>.

* + Поставьте мышку на имя поля <Предметы> в таблице <Преподаватели>

, и не отпуская кнопку мыши перетащите его на поле <Любимый предмет> таблицы

<Личные данные>. Отпустите мышку. Появиться диалоговое окно <Связи>, в кото- ром включите значки «*Обеспечение целостности данных*», «*Каскадное обновление связанных полей*» и «*Каскадное удаление связанных полей»*. Щелкните по кнопке <Со- здать>. Появиться связь «один-ко-многим».

* + Поставьте мышку на имя поля <Код студента> в таблице <Личные дан- ные> и перетащите его, не отпуская мышки, на поле <Код> таблицы <Ведомость успевае- мости>. В появившемся окне <Связи> включите значок «*Обеспечение целостности дан- ных*» и щелкните по кнопке <Создать>. Появиться связь «один-к- одному».
  + Закройте схему данных, сохранив ее.

1. Произведите фильтрацию данных в таблице <Личные данные> *по выделен-*

*ному*.

* Откройте таблицу в режиме таблицы.
* Выберите студентов, проживающих в Новороссийске. Для этого по-

ставьте курсор в одну из первых записей, где есть город Новороссийск и щелкните по кнопке

- *Фильтр по выделенному* на панели инструментов. Выберите команду <Равно

«Новороссийск» >. Access отобразит все записи, удовлетворяющие критерию филь- трации.



* + Для отображения всех записей выполните команду <Удалить фильтр> для этого щелкните по соответствующей кнопке на панели инструментов .

1. Закончите работу с базой данных Access.

## Практическая работа № 7 Редактирование, добавление и удаление записей в таблице. Применение логических условий к записям. Открытие, редактирование и

**пополнение табличного файла**

**Цели работы:** закрепить навыки по редактированию таблиц; познакомиться с ос- новными видами запросов; научиться создавать запросы на выборку различными спосо- бами; научиться создавать сложные запросы; научиться создавать перекрестные запросы.

**Запрос** – это средство, с помощью которого извлекается из базы данных информа- ция, отвечающая определенным критериям. Результаты запроса представляют не все записи из таблицы, а только те, которые удовлетворяют запросу.

Запросы состоят из ряда условий, каждое условие состоит из трех элементов:

1. поле, которое используется для сравнения;
2. оператор, описывающий тип сравнения;
3. величина, с которой должно сравниваться значение поля. Выражения и опера- торы, применяемые в условиях отбора.

|  |  |
| --- | --- |
| Выражения и операторы | Описание выражений и операторов |
| **Числа** | **Вводятся без ограничений** |
| Текст | Должен быть заключен в кавычки |
| Даты | Ограничиваются с двух сторон символами # (например, #01.02.02#) |
| \*; +; -; /; ^ | Арифметические операторы, связывающие выражения |
| <; <=; >; >=; =; <> | Операторы сравнения |
| And (И); Not (Нет); Or (Или) | Логические операторы |
| Like | Используется для логики замены в выражениях |
| In | Для определения, содержится ли элемент данных в списке  значений |
| Between… And… | Для выбора значений из определенного интервала |
| ? | Заменяет один символ (букву или цифру) |
| \* | Заменяет несколько символов |

Запросы могут быть простые, сложные перекрестные.

## Задание для практической работы Создайте запросы к вашей базе данных

* 1. Откройте свою учебную базу данных.
  2. Создайте запрос на выборку студентов, у которых по всем предметам только хорошие оценки с помощью *Мастера запросов*.
     + На панели инструментов выберите команду <Мастер запросов>.
     + В появившемся диалоговом окне выберите <Простой запрос> и щелкните по кнопке <OK>.
     + В следующем окне выберите таблицу, по которой строится запрос (<Ведомость успеваемости>), и те поля, которые участвуют в запросе. Перенесите их в правую часть окна с по- мощью кнопки нажмите <Далее>. В следующем окне тоже нажмите <Далее>.

,



* + - В другом окне дайте название запроса «Хорошисты» и нажмите <Готово>.
    - Появится таблица <Хорошисты>, в которой отражены фамилии всех студентов и изучаемые предметы.
    - Откройте таблицу «Хорошисты», перейдите в режим <Конструктор>. Здесь в

поле

<Условия отбора> под каждым предметом поставьте условие >=4 или 4OR5. Примечание: Галочки в каждом поле означают, что по вашему выбору можно вклю-

чить или убрать любое поле на выборку.

* + - Перейдите в режим таблицы, ответив <Да> на вопрос о сохранении запроса. (В таблице должны остаться фамилии «хорошистов»).
  1. С помощью <Конструктора запросов> создайте запрос на выборку по таблице

<Личные данные>.

* + - Щелкните по таблице <Личные данные>, зайдите в меню <Создание>, вы- берите команду <Конструктор запросов >.
    - Добавьте нужную таблицу в поле запроса. Выделите еѐ в списке и щелкните по кнопке <Добавить>. Закройте окно <Добавление таблицы>.
    - Выберите студентов, чьи фамилии начинаются на букву «В» и которые прожи- вают в Анапе. Для этого:
* добавьте в строку <Поле> два поля <Фамилия> и <Город>;
* в строке <Условия отбора> в первом столбце укажите значение

Like ―B \* ‖, а во втором столбце с названием <Город> - «Анапа»;

* закройте запрос, сохранив его под названием ―ВВВ‖ (у вас должны остаться в списке студенты, проживающие в Анапе). Рисунок 7.1.

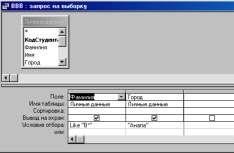


Рисунок 7.1. Запрос на выборку

Индивидуальные задания к практической работе

а) Составьт запрос с названием <Запрос 1> на базе таблицы <Ведомость успеваемо- сти>, в котором будут указаны студенты, имеющие по первым двум предметам оценки не менее «4». (Выполните запрос или через Конструктор запросов, или через Мастер запросов) б) Составьте <Запрос 2> на базе таблицы <Ведомость успеваемости>, в котором бу-

дут указаны студенты, имеющие не более 30 часов пропусков по неуважительной причине. Добавьте в этот запрос поле пропуски по уважительной причине в интервале от 30 часов до 45 часов (используйте оператор Between… And…)

в) Составьте <Запрос> на базе таблицы <Личные данные>. Выведите список студен- тов, которым на данный момент, т.е. на сегодняшнее число, исполнилось уже 17 лет (ис- пользуйте оператор Between… And…)

Примечание: Дата записывается с использованием символа #, например, #01.02.02.# Составьте запрос на базе трех таблиц <Ведомость успеваемости>, <Личные данные>

и

<Преподаватель>. Выберите студентов, которые проживают в Новороссийске и у ко-

торых любимый предмет «Менеджмент». Озаглавьте <Запрос 4>. Используйте <Конструк- тор запросов>.

В меню <Создание> выберите <Конструктор запросов>.

Добавьте все три таблицы в поле запроса. Закройте окно <Добавление таблицы>.

В первый столбец в строку <Поле> перетащите из первой таблицы с помощью мышки <Фамилия>, из второй таблицы во второй столбец <Город> и из третей таблицы в третий столбец строки <Поле> - <Предмет> (Рисунок 7.2).

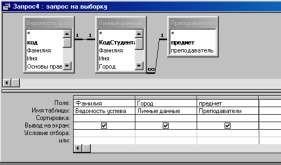


Рисунок 7.2. Запрос на выборку

В поле <Условия отбора> в столбце <Город> введите город «Новороссийск», в стол- бец <Предмет> введите «Менеджмент».

Сохраните запрос под именем <Запрос 4>.

Откройте запрос и проверьте результат проделанной работы.

Выполните запрос на создание новой таблицы, в которой должны быть поля

<Фамилия>, <Имя>, <Пропуски по неуважительной причине>, <Город> и <Пред-

мет>.

В меню <Создание> выберите <Конструктор запросов>.

Добавьте все три таблицы из списка окна <Добавление таблицы>. Закройте это окно. В первую строчку <Поле> из первой таблицы перенесите в первый столбец поля<Фа-

милия>, во второй <Имя> и в третий <Пропуски по уважительной причине>, в четвертый столбец перетащите поле <Город> из второй таблицы и в последнем столбце будет поле

<Предмет> из третьей таблицы.

Закройте запрос, сохранив его с именем <Запрос 5>. Создайте перекрестный запрос.

Допустим, нужно посчитать для ведомости, сколько в группе человек получили по предмету ―троек‖, ―четверок‖ и ―пятерок‖. Для этих целей используется перекрестный запрос.

В меню <Создание> выберите <Мастер запросов>.

В диалоговом окне выберите <Перекрестный запрос>, щелкните по кнопке <OK>. В окне <Создание перекрестных запросов> выделите таблицу <Ведомость успевае-

мости> и щелкните <Далее>.

Выберите поля, значения которого будут использоваться в качестве заголовок строк – это <Фамилия> и <Имя>. Щелкните по кнопке <Далее>.

Выберите поле, значение которого будут использоваться в качестве заголовков столбцов, например <Менеджмент>. Щелкните по кнопке <Далее>.

Выберите функцию, по которой будут вычисляться значения ячеек на пересечении столбцов и строк (в данном случае Count – количество). Щелкните по кнопке <Далее>.

Задайте имя запроса <Итог по менеджменту> и щелкните по кнопке <Готово>. Со- ставьте аналогичные запросы для оценок по трем другим предметам.

Предъявите преподавателю все запросы своей базы данных на экране дисплея. Завершите работу с Access.

формы.

## Практическая работа №8 Проведение сортировки и фильтрации данных. Поиск данных по одному и нескольким полям. Поиск данных в таблице

**Цели работы:** Научиться создавать формы ввода-вывода; научиться создавать кно- почные

**Форма** – это средство, упрощающее ввод, редактирование и отображение инфор-

мации,

хранящейся в таблицах базы данных. Она представляет собой окно с набором эле-

ментов управления.

Форма сама по себе не хранит информацию, она просто обеспечивает удобный спо- соб доступа к информации, хранящейся в одной или нескольких таблицах. Формы по срав- нению с обработкой данных в режиме таблицы обладают следующими преимуществами:

* + - Форма позволяет в каждый момент сфокусировать внимание на отдельной записи;
    - Элементы управления на форме можно расположить логичным образом, об- легчающим чтение и работу с данными;
    - Отдельные элементы управления обладают возможностями облегчить ввод и изменение отдельных данных;
    - Некоторые объекты баз данных, такие как рисунки, анимации, звуки и видеоклипы, могут отображаться только в режиме формы, но не в режиме таблицы.

Создание кнопочной формы.

Кнопочное меню представляет собой форму, на которой расположены элементы управления – кнопки с поясняющими надписями. Щелчок на кнопке открывает соответ- ствующую таблицу, запрос, форму или отчет. Меню - удобный инструмент работы с базами данных, и он практически всегда присутствует в базах созданных для предприятий или фирм.

Кнопочное меню создают с помощью Диспетчера кнопочных форм.

Отчет – это гибкое и эффективное средство для организации просмотра и распечатки итоговой информации. В отчете можно получить результаты сложных расчетов, статисти- ческих сравнений, а также поместить в него рисунки и диаграммы. Пользователь имеет воз- можность разработать отчет самостоятельно (в режиме *Конструктора*) или создать отчет с помощью *Мастера*, т.е. полуавтоматически.

## Задание для практической работы Задание 1. Создайте формы к базе данных

1. Откройте свою базу данных.
2. Создайте форму с помощью <Мастера форм> на базе таблицы

<Ведомость успеваемости>.

* + Откройте таблицу <Ведомость успеваемости>.
  + Выберите закладку <Формы >, щелкните мышкой по кнопке <Другие формы>.
  + В появившемся диалоговом окне выберите <Мастер форм>.
  + В поле <Таблицы/Запросы> выберите таблицу <Ведомость успеваемо- сти>, в поле

<Доступные поля> выберите поля <Фамилия>, <Имя> и перенесите их стрелкой в

поле

<Выбранные поля>. Также перенесите поля с названием предметов, щелкните по

кнопке <Далее>.

* + Выберите внешний вид формы – Табличный, щелкните по кнопке <Далее>.
  + Выберите требуемый стиль (н-р, Обычная), щелкните по кнопке <Далее>.
  + Задайте имя формы <Успеваемость> и щелкните по кнопке <Готово>. В результате получите форму, в которой можно менять данные и вводить новые значения.
  + Закройте форму.

1. Создайте форму на основе таблицы <Преподаватели>.
   * Откройте таблицу <Преподаватели>.
   * Выберите закладку <Формы >, щелкните мышкой по кнопке <Другие формы>.
   * В появившемся диалоговом окне выберите <Мастер форм> .
   * Выберите внешний вид формы - < ленточный>.
   * Выберите любой стиль.
   * Получите готовую форму. Сохраните ее под именем <Преподаватели>.
   * Закройте форму.
   * Создайте форму <Личные данные> с помощью инструмента <Пустая форма>
   * На вкладке Создание в группе Формы щелкните Пустая форма. 
   * Access открывает пустую форму в режиме макета и отображает область Спи- сок полей.
   * В области Список полей щелкните знак плюс (+) рядом с таблицей или таб- лицами, содержащими поля, которые нужно включить в форму.
   * Чтобы добавить поле к форме, дважды щелкните его или перетащите его на форму. Чтобы добавить сразу несколько полей, щелкните их последовательно, удерживая нажатой клавишу CTRL. Затем перетащите выбранные поля на форму.
   * Закройте окно списка полей.
   * Перейдите в режим Конструктора

**Примечание 1.** Размер окошка для названия поля и для его значений меняются мыш-

кой.

Для этого выделите черный квадратик рамки (рамка станет цветной), установите

курсор на границу рамки и с помощью двунаправленной стрелки измените размеры рамки.

**Примечание 2.** С помощью кнопок панели инструментов Шрифт меняйте соответ- ственно цвет фона, текста, линии/границы и т.д.

* + Расположите элементы удобно по полю.
  + Задайте размер текста поля <Фамилия> равным 24 пт, шрифт - синего цвета.
  + Увеличьте в высоту рамку поля <Фотография>.
  + Сохраните форму с именем <Данные студентов>.
  + Посмотрите все способы представления форм: в режиме Конструктора, ре- жиме Макета и режиме Форм.
  + Закройте форму.

1. Добавьте в таблицу <Личные данные> логическое поле <Инсти- тут> (т.е., собирается ли в дальнейшем учащийся поступать в институт). Значение этого поля <ДА> или

<НЕТ>.

* + Откройте таблицу <Личные данные> в режиме Конструктор. Добавьте поле с именем

<Институт> и типом Логический. Закройте таблицу.

* + Перейдите на закладку Формы и откройте форму <Данные студентов> в режиме Конструктор
  + Щелкните по кнопке <Список полей> на панели инструментов, выделите название

<Институт> и перетащите его мышкой в область данных, появиться зна-

чок



и надпись <Институт>.

* + Расположите новые элементы по правилам оформления формы (с помощью мыши).
  + Закройте <Список полей>

**Примечание 3.** Если флажок установлен, поле в таблице имеет значение

<ДА>, если снят, то <НЕТ>.

* + Перейдите в режим <Раздельная форма> и посмотрите записи. Установите флажки у восьми разных учащихся.
  + Закройте форму, ответив утвердительно на вопрос о сохранении.

1. Создайте кнопочную форму <Заставка> с помощью Конструктора.
   * Щелкните по кнопке <Создать>.
   * Выберите <Конструктор>. Появиться пустая форма. Задайте мышкой ши- рину формы, равную 10см, а высоту – 7см.
   * Сохраните работу с именем <Заставка>.
   * Откройте созданную форму <Заставка> в режиме Конструктора.
   * Выберите на панели инструментов <Элементы управления> кнопку Аа –

<Надпись>. Курсор мышки примет вид крестика с «приклеенной» буквой А. Щелкните мышкой по месту начала надписи и введите:

*База данных*

*«Гостиница» группа ГС - 31*

(после слов База данных нажмите одновременно комбинацию клавиш Shift+Enter.)

* + Нажмите клавишу <Enter>. Выберите размер букв 18, а выравнивание - по центру. Цвет фона – голубой. Растяните мышкой надпись на ширину окна.
  + Выберите на панели элементов значок - Кнопка. Щелкните мышкой по тому месту области данных, где должна быть кнопка. Появиться диалоговое окно <Созда- ние кнопок>.



* + Выберите категорию <Работа с формой>, а действие <Открыть форму>, и щелкните по кнопке <Далее>.
  + Выберите форму <Успеваемость> , открываемую этой кнопкой щелкните по кнопке

<Далее>. В следующем окне также щелкните по кнопке <Далее>.

* + В следующем окне поставьте переключатель в положение <Текст>, набе- рите в поле слово <Успеваемость> (Рисунок8.1) и щелкните по кнопке <Далее>.

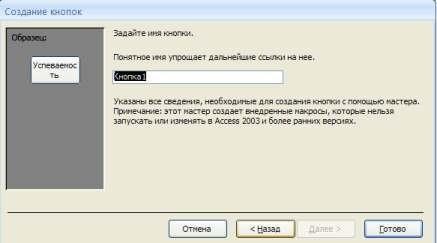


Рисунок 8.1. Создание кнопок

* + Задайте имя кнопки <Успеваемость> и щелкните по кнопке <Готово>.

**Примечание 4.** Размер и расположение кнопок можно менять мышкой в режиме Конструктор.

Самостоятельно создайте кнопки для форм <Личные данные> и <Преподаватели>.

* + Перейдите в режим формы (Рисунок 8.2). Теперь при щелчке мы- шью по соответствующим кнопкам будут открываться соответствующие формы для ра- боты.
  + Закройте форму.

Рисунок 8.2. Окно формы

1. Создайте кнопочную форму при помощи Диспетчера кнопочных форм.
   * Откройте вкладку Работа с базами данных, команда - Диспетчер кнопочных форм. Вы получите диалоговое окно, представленное на Рисунке 8.3.

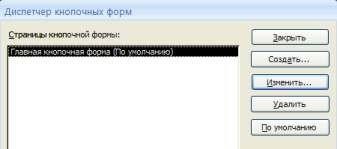


Рисунок 8.3. Диспетчер кнопок

* + Щелкните в этом окне по кнопке <Изменить>.
  + В следующем окне щелкните по кнопке <Создать> и в появившемся окне измените содержимое полей в соответствии с Рисунком 8.4 (Команду и Форму выбирайте из списка, а не набирайте вручную). Щелкните по кнопке <ОК>.

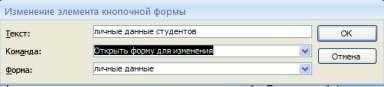


Рисунок 8.4. Изменение элементов кнопочной формы

* + Аналогично создайте еще три элемента кнопочной формы: <Успе- ваемость>,

<Преподаватели > и <Заставка>.

* + Добавьте кнопку закрытия базы данных. Для этого щелкните по кнопке

<Создать>, наберите в поле Текст слово <Выход>, а в поле Команда выберите <Выйти из приложения>. Закройте диалоговые окна.

* + Откройте окно <Кнопочная форма> в режиме Конструктора или Макета, из- мените цвет надписи и название вашей базы данных на ГОСТИНИЦА, сохраните форму.
  + Украсьте вашу форму рисунком. Для этого щелкните по значку Эмблема и

выберите в открывшемся окне папку с рисунками, выберите понравившийся и вставьте в свою кнопочную форму.

* + Перейдите в режим формы, проверьте работу всех кнопок кнопочной формы. Завершите работу с базой данных, нажав на кнопку <Выход>.

**Задание 2. Создайте отчет с помощью *Мастера отчетов.***

* + Откройте вкладку *Создание,* меню *Отчеты*.
  + Выберите *Мастер отчетов* и таблицу «Личные данные».
  + Выберите нужные поля, которые будут участвовать в отчете, нажмите кнопку «Далее».
  + В новом окне выберите поля для группировки так, чтобы сначала было указано поле

«Фамилия», нажмите кнопку «Далее».

* + На этом шаге отсортируйте данные по алфавиту, нажмите кнопку «Далее».
  + Выберите вид макета *Ступенчатый* и щелкните по кнопке «Далее».
  + Выберите стиль отчета: *Открытая* и щелкните по кнопке «Далее».
  + Задайте имя отчета: «Отчет1» и щелкните по кнопке «Готово». Вы попадете в режим просмотра отчета.
  + Закройте отчет согласившись с сохранением.

Самостоятельно. Составьте еще два отчета по запросам – «Запрос 3» и «Запрос 5», выбирая из разных макетов: *блок*; *структура*, выбирая из разных стилей. Сохраните отчеты под именами «Отчет 2» и «Отчет 3».

**Задание 3. Создайте Пустой отчет** в столбец на базе таблицы «Ведомость успевае- мости» и сохраните его с именем «Успеваемость».

С помощью Конструктора измените цвет букв заголовка, их размер и шрифт.

### Задание 4. Создайте почтовые наклейки.

* + Откройте вкладку *Создание,* меню *Отчеты*.
  + Выберите таблицу «Личные данные», команда Наклейки.
  + В следующем окне щелкните по кнопке «Далее».
  + В следующем окне выберите шрифт, размер шрифта, насыщенность и цвет, вновь щелкните по кнопке «Далее».
  + В следующем окне создайте прототип наклейки, напечатав слово ЛИЧ- НОСТЬ и выбрав соответствующие поля, щелкните по кнопке «Далее».
  + В следующем окне укажите поля для сортировки (Фамилия, Имя), щелкните по кнопке

«Далее».

* + Введите имя отчета «Наклейки» и щелкните по кнопке «Готово».
  + Просмотрите Наклейки (Рисунок 8.5).

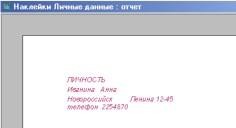


Рисунок 8.5. Отчет

## Практическая работа №9 Работа с переменными. Написание программного файла и работа с табличными файлами. Заполнение массива из табличного файла.

**Заполнение табличного файла из массива**

**Цель работы:** овладение навыками работы с переменными, обработка табличных файлов в среде VisualFoxPro.

# VisualFoxPro

При создании приложения используется проект, который объединяет элементы при- ложения VisualFoxPro и группирует их по типам. База данных в VisualFoxPro – это сово- купность таблиц, отношений между таблицами, индексов, триггеров, хранимых процедур. База данных является частью проекта, поэтому еѐ целесообразно создавать в окне проекта. ***Структурными элементами базы данных являются:***

Поле – элементарная единица логической организации данных, которая соответ- ствует неделимой единице информации – реквизиту. Для описания поля используются сле- дующие характеристики: имя, длина, тип и точность (для числовых данных). В структуре записи файла указываются поля, значения которых являются ключами: первичными и внешними.

Первичный ключ - это одно или несколько полей, однозначно идентифицирующих запись. Если первичный ключ состоит из одного поля, он называется простым, если из не- скольких полей - составным ключом.

Внешний ключ - это одно или несколько полей, которые выполняют роль поисковых или группировочных признаков. В отличие от первичного, значение внешнего ключа может повторяться в нескольких записях файла, то есть он не является уникальным. Если по зна- чению первичного ключа может быть найден один единственный экземпляр записи, то по внешнему – несколько.

Файл (таблица) – совокупность экземпляров записей одной структуры. Описание ло- гической структуры записи файла содержит последовательность расположения полей за- писи и их основные характеристики,

Запись – совокупность логически связанных полей.

Экземпляр записи - отдельная реализация записи, содержащая конкретные значения ее полей.

Таблицы составляют основу вашей базы данных. В них будет храниться вся необходимая информация. В дальнейшем данные в таблице будут дополняться новыми данными, редак- тироваться или исключаться из таблицы. Поля таблицы предназначены для хранения в них данных. Это могут быть числа, текстовая информация, даты, графические файлы и т.д. В VisualFoxPro допустимыми являются типы полей, указанные в таблице 9.1.

Таблица 9.1.Некоторые типы данных, используемых в СУБД FoxPro



Таблица 9.2. Некоторые команды СУБД FoxPro



# Задание для практической работы

Создать в служебной папке Мои документы новую папку и присвоить ей имя, пример: 3курс4группаfoxlab (указывать свой курс и группу)

Запустить программу MicrosoftVisualFoxPro: Пуск/программы/VisualFoxPro

Ознакомиться с элементами рабочего окна программы

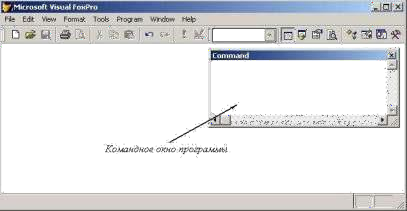
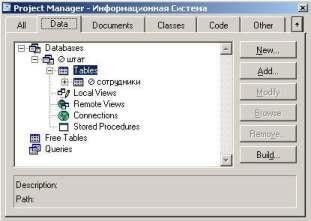


Рисунок 9.1. Главное окно VisualFoxPro 8.0

Создать новый проект: File/New/Project/NewFile, указать созданную нами ранее папку, присвоить имя проекту Информационные системы и сохранить. Все создаваемые в последующем элементы приложения будут храниться в проекте Информационные системы БД создаѐтся аналогично: File/New/Database/Newfile, присвоить имя Штат исохра-

нить. Структура проекта и его элементы отражаются в окне программы ProjectManager (Ме-

неджер проекта)

Рисунок 9.2. ОкноProject Manager

Добавить БД в проект: в окне ProjectManager щѐлкнуть на вкладке Data/Databases/Add, в открывшемся диалоговом окне выбрать БД и нажать OK

Создать таблицу в БД штат: в окне ProjectManager (см. рис.9.2.) щѐлкнуть клавишей мыши на вкладке Data/Databases/штат/Tables/New/Newtable, присвоить имя Сотрудники и сохранить

В появившемся окне TableDesigner(Конструктор таблиц) (см. рис.9.3.) на вкладке Fields (Поля) создать структуру таблицы в соответствии с таблицей 3.

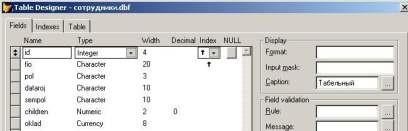
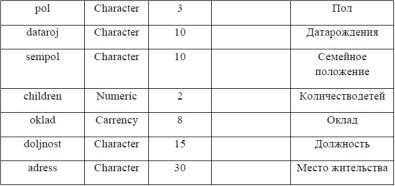


Рисунок 9.3. Окно конструктора таблицы TableDesigner вкладка Fields

Таблица 9.3.Определение полей таблицы Сотрудники в окне конструктора таблицы TableDesigner



Перейти на вкладку Indexes(Индексы) окна конструктора таблицы TableDesigner и присвоить созданному индексу значения в соответствии с таблицей 9.4. Это необходимо для создания ключевого поля



Для ввода и редактирования данных в таблице *Сотрудники* в командном окне про- граммы введите команду APPEND и нажмите клавишу Enter Командное окно программы **FoxPro** (см. рис.9.1.) предназначено для ввода команд с клавиатуры и последующего их выполнения. Любые действия и операции над элементами приложения в **СУБД FoxPro** мо- гут осуществляться при помощи команд, вводимых в программное окно.

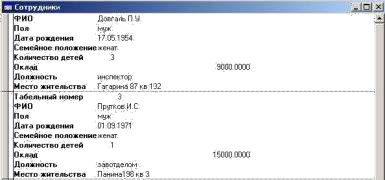


Рисунок 9.4. Просмотр и редактирование содержимого таблицы в режиме APPEND

После ввода данных (см. таблицу 9.4.) в командном окне программы введите команду BROWSE и нажмите клавишу Enter. Результат должен соответствовать рис.9.5.

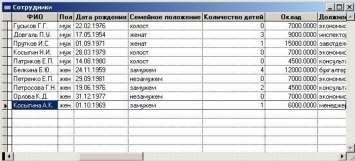
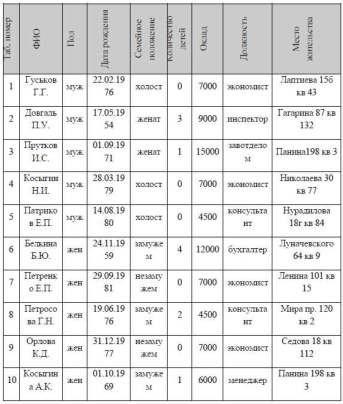


Рисунок 9.5. Просмотр таблицы в режиме BROWSE Таблица 9.4. Содержимое таблицы Сотрудники



В окне BROWSE установите курсор в нижней части таблицы и нажмите комбинацию клавиш Ctrl + Y на клавиатуре. В таблицу добавится новая строка

Прежде чем удалить строку в VisualFoxPro еѐ сначала необходимо пометить на уда- ление. Для этого щелкнуть клавишей мыши в ячейке слева от удаляемой записи в области узкого, непоименованного столбца, ячейка станет помеченной чѐрным цветом, или выде- лить необходимую строку и выбрать команду ToggleDeletionMark (Установка метки на уда- ление) пункта Table (Таблица) системного меню VisualFoxPro, запись будет помечена на удаление. Удалить запись, выбрав команду RemoveDeletedRecords (Удалить запись) пункта Table (Таблица) системного меню VisualFoxPro. Запись будет удалена

Для закрытия всех элементов приложения в командном окне программы введите команду CLOSE ALL

Для завершения работы с программой **VisualFoxPro** в командном окне программы введите команду QUIT.

## Создание отношений между таблицами в многотабличной БД Библиотека

Среди требований, предъявляемых к СУБД, основное место занимает возможность быстрого поиска необходимой информации. Средством, позволяющим решить эту про- блему, является применение индексов. В VisualFoxPro для создания первичных ключей, определяющих отношения между таблицами и условия целостности данных также предна- значены индексы. В этом случае индексы должны быть уникальными, то есть значения ин- дексированного поля должны быть неповторяющимися (уникальными).

Для создания индекса таблицы используется вкладка Indexes (Индексы) окна кон- структора таблиц TableDesigner. Все индексы в VisualFoxPro имеют имена, задаваемые в поле Name (Имя). Для задания типа создаваемого индекса используется список Type (Тип). Таблица 9.10.Описание типов индекса



Обычно, в VisualFoxPro при создании форм, отчѐтов и запросов используются не- сколько таблиц, между которыми установлены постоянные отношения. Такие таблицы называются связанными. Из двух связанных таблиц одна является главной (родительской), а другая подчинѐнной (дочерней). При создании индексов для родительской таблицы дол- жен быть определѐн ключ типа Primary (Первичный) или типа Candidate (Кандидат), а для дочерней таблицы – индекс для связи с родительской таблицей типа Regular (Обычный).

Таблица 9.11. Типы отношений между таблицами



Одним из самых важных требований, предъявляемых к базам данных, является це- лостность данных, которую определяют установленные между таблицами отношения. Для

определения целостности данных в VisualFoxPro используется окно построителя условий

целостности данных ReferentialIntegrityBuilder (Построитель целостности данных), которое содержит перечень всех установленных отношений между таблицами (см. рис.9.3.).

Таблица 9.12.Описание действий VisualFoxPro, в зависимости от выбранной оп- ции, при изменении значения первичного ключа или ключа типа

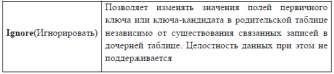
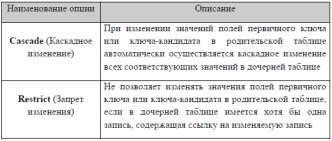


Таблица 9.13.Действия VisualFoxPro, в зависимости от выбранной опции, при удалении записи из родительской таблицы



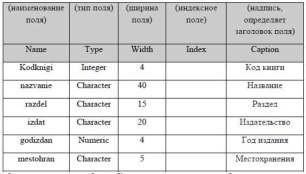
Таблица 9.14.Действия **VisualFoxPro**, в зависимости от выбранной опции, при до- бавлении новой записи в родительскую таблицу



## Индивидуальные задания к практической работе

Создать многотабличную БД и присвоить имя Библиотека. Создать следующие таблицы:

1. Таблица Книги



Определение полей таблицы Книги в окне конструктора таблицы **TableDesigner**

Содержимое таблицы Книги



1. Таблица Разделы



Определение полей таблицы Разделы в окне конструктора таблицы TableDesigner Содержимое таблицы Разделы



1. Таблица *Издательство*

Определение полей таблицы *Издательство* в окне конструктора таблицы **TableDesigner**



Содержимое таблицы *Издательство*

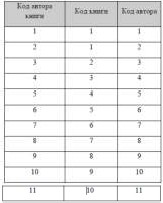


1. Таблица *Автор\_книги*

Определение полей таблицы Автор\_книги в окне конструктора таблицы **TableDesigner**



Содержимое таблицы *Автор\_книги*

**Таблица *Авторы*

Определение полей таблицы *Авторы* в окне конструктора таблицы **TableDesigner**



**Содержимое таблицы *Авторы*

1. Открыть БД штат: в окне ProjectManager (Менеджер проекта) вкладка

Data/Databases/Библиотека/Open

1. Необходимо модифицировать таблицы БД Библиотека, создать индексирован- ные поля, а также первичные ключи для осуществления связей между таблицами. Для мо- дификации (изменения структуры) таблицы в окне проекта ProjectManager (Менеджер про- екта) установить курсор на модифицируемую таблицу Книги и нажать кнопку Modify (Мо- дифицировать) или в окне конструктора БД DatabaseDesigner (Конструктор базы данных) установить курсор в таблицу Книги, вызвать контекстное меню и выбрать команду Modify

(Модифицировать). На экране откроется диалоговое окно TableDesigner (Конструктор таб- лицы)

1. В окне TableDesigner (Конструктор таблицы) перейти на вкладку Indexes (Ин- дексы) и ввести значения в соответствии с таблицей 6., нажать кнопку OK, система попро- сит подтвердить сохранение изменений, нажать Yes

Таблица 9.15.Определение индексов таблицы Книги на вкладке Indexes (Индексы) окна конструктора таблицы TableDesigner

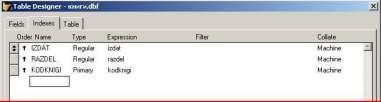
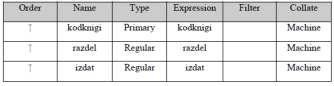


Рисунок 9.5. Вкладка Indexes (Индексы) окна TableDesigner таблицы Книги Индекс, с при- своенным типом Primary является первичным ключом таблицы

1. Аналогично внести изменения в таблицы БД Библиотека: Разделы,

Издательство, Автор\_книги, Авторы в соответствии с таблицами 9.16,9.17, 9.18, 9.19 Таблица 9.16.Определение индексов таблицы Разделы на вкладке Indexes (Ин-

дексы) окна конструктора таблицы TableDesigner



Таблица 9.17.Определение индексов таблицы Издательство на вкладке Indexes (Индексы) окна конструктора таблицы TableDesigner



Таблица 9.18.Определение индексов таблицы Автор\_книги на вкладке Indexes (Индексы) окна конструктора таблицы TableDesigner

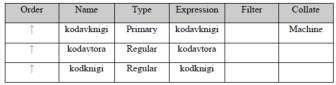


Таблица 10. Определение индексов таблицы Авторы на вкладке Indexes (Ин-



дексы) окна конструктора таблицы TableDesigner

1. В окне DatabaseDesigner (Конструктор базы данных) выбрать родительскую таб- лицу Разделы. Таблицы в конструкторе БД обозначаются прямоугольниками, в нижней части кото- рых после надписи Indexes (Индексы) расположен список индексов, созданных для данной таблицы.

Первичный ключ в этом списке выделяется значком ключа, расположенным с левой стороны от наименования индекса. Установить курсор мыши на первичный ключ razdel таблицы Разделы. Нажать кнопку мыши и, не отпуская ее, переместить курсор мыши на индекс razdel дочерней таб- лицы Книги. Отпустить кнопку мыши. Созданные отношения между таблицами отображаются в виде линий. Аналогичным образом, создать связь между таблицами Издательство и Книги (Таблица Издательство – родительская, таблица Книги – дочерняя, первичный ключ – izdat)

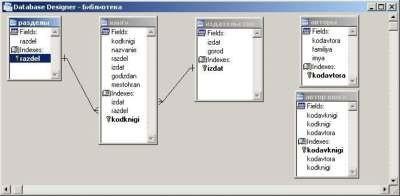


Рисунок 9.6. Отображение связей между таблицами в окне DatabaseDesigner (Конструктор базы данных)

Для редактирования отношений между связанными таблицами дважды щелкнуть левой кнопкой мыши на линии, появится диалоговое окно EditRelationship (Редактирование от- ношения), в котором слева приведено наименование родительской таблицы и расположен раскрывающийся список индексов таблицы, справа размещена аналогичная информация о дочерней таблице. В этом диалоговом окне указан также тип установленного отношения между таблицами.

В окне EditRelationship(Редактирование отношения) нажать кнопку ReferentialIntegrity(Це- лостность данных), в появившемся диалоговом окне ReferentialIntegrityBuilder (Построи- тель целостности данных) выбрать отношение издательство- книги. В полях Update (Изме- нить), Delete (Удалить), Insert (Заменить) установить тип действий Restrict (Запрет измене- ния). Провести аналогичные действия для отношения разделы-книги. Результат описан- ных действий, которые необходимы для обеспечения целостности данных, представлен на рис.9.7.

Для сохранения выполненных действий нажать кнопку OK, система потребует подтвер- ждение сохранения, нажать кнопку Да

Нажать кнопку OK для закрытия диалогового окна EditRelationship

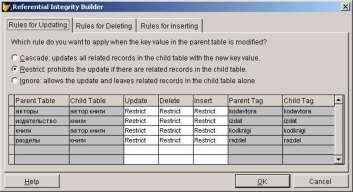


Рисунок 9.7. Диалоговое окно ReferentialIntegrityBuilder(Построитель целостности дан- ных)

Создать связь между таблицами Книги и Автор\_книги (kodknigi) Создать связь между таблицами Авторы и Автор\_книги (kodavtora)

Установить целостность данных в созданных отношениях: в полях Update (Изменить), Delete (Удалить), Insert (Заменить) установить тип действий Restrict (Запрет изменения)

## Практическая работа №10 Создание формы. Управление внешним видом формы

**Цель работы**: овладение практическими навыками по созданию форм. Формы в VisualFoxPro

В VisualFoxPro для просмотра, ввода и редактирования данных, хранящихся в таб- лицах, используются формы, являющиеся более наглядным средством представления ин- формации. Важным преимуществом форм является, то, что они позволяют работать не с одной, а с несколькими связанными таблицами, что, в свою очередь, также увеличивает наглядность.

При создании форм в VisualFoxPro разработчик может использовать следующие средства: FormWizard (мастер форм), FormBuilder (построитель формы), Builder (построи- тель объектов формы), AutoFormatBuilder (построитель автоформата), FormDesigner (кон- структор форм).

Чтобы создать форму для одной или связанных таблиц с возможностью задания отображаемых в форме полей, стиля их отображения и указания типа кнопок управления, можно использовать FormWizard (мастер создания форм).

Для самостоятельной разработки формы с заданными свойствами или изменения формы, созданной с помощью мастера, вам необходимо использовать FormDesigner (кон- структор форм).

Для облегчения размещения в конструкторе форм полей и надписей, оформленных в соответствии с выбранным стилем, можно использовать FormBuilder (Построитель формы).

Создание формы с помощью конструктора форм

Любая форма в VisualFoxPro состоит из объектов, каждый из которых имеет харак- терные свойства. Для любого объекта вы можете указать действия, выполняемые написан- ной разработчиком программой при наступлении определѐнных событий. Процесс созда- ния формы в конструкторе форм состоит в размещении в форме объектов и определении свойств, а также связанных с ними событий и выполняемых действий.

Процесс создания формы включает следующие действия: настройка параметров формы

определение среды окружения, то есть выбор используемых в форме таблиц и уста- новка связей между ними

размещение в форме объектов: текста, полей различных типов, линий, рисунков, кнопок управления.

Задание для практической работы

Задание 1. Создание формы средствами мастера форм Запустить программу MicrosoftVisualFoxPro

Открыть созданный проект: File/Open/Информационная система

Открыть БД штат: в окне ProjectManager (Менеджер про- екта) вкладка

Data/Databases/штат/Open

Создать форму Сотрудники, щелкнув клавишей мыши на вкладке Documents/Forms/New/FormWizard/FormWizard, нажать клавишуOk



Рисунок 10.1. Диалоговое окно WizardSelection

В появившемся окне FormWizard в области Databasesandtables (Базы данных и таблицы) выбрать необходимую БД штат и указать таблицу Сотрудники, для которой создается форма. Из области

Availablefields (Имеющиеся поля) выбрать поля, которые будут размещены в форме, в со- ответствии с рис.10.2, используя кнопки расположенные между списками для перехода к следующему шагу нажать кнопку Next(Далее)

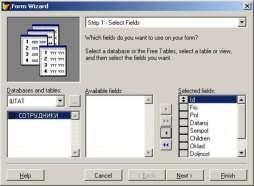


Рисунок 10.2. Диалоговое окно выбора полей для отображения.

В появившемся диалоговом окне в области Style установить стиль отображения объектов и в области Button type типы кнопок управления в соответствии с рис.10.3, нажать кнопку Next

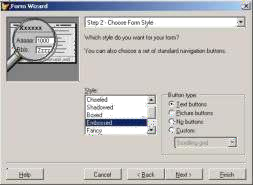


Рисунок 10.3. Окно выбора стиля отображения полей и управляющих кнопок.

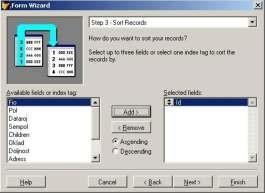
В появившемся диалоговом окне задать критерий сортировки данных, указав поля по ко- торым будет осуществляться упорядочивание в соответствии рис.10.4. нажать кнопку Next

Рисунок 10.4. Установка критерия упорядочения данных

На заключительном шаге создания форм в области Typeatitleforyourform(Тип заголовка формы) указать имя формы

Сотрудники. Воспользовавшись кнопкой Preview (Просмотр) просмотрите, как будет вы- глядеть создаваемая форма, после просмотра нажмите кнопку ReturntoWizard.



Рисунок 10.5.Окно задания заголовка формы.

Если что-то не так, вернитесь к предыдущим шагам, воспользовавшись кнопкой Back. Нажать кнопку Finish (Готово) и в появившемся диалоговом окне SaveAs (Сохранить как) указать папку, в которой будет размещена форма Сотрудники

Запустить в окне ProjectManager форму Сотрудники: Documents/Forms/Сотрудники/Run

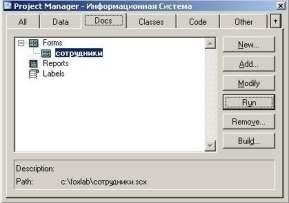


Рисунок 10.6. Окно ProjectManager

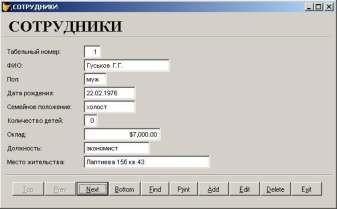
Или с помощью меню программы VisualFoxPro: Program/Doзадать имя формы Сотрудни- кии тип файлаForm, нажать кнопкуDo. С помощью кнопок (см. таблицу 10.1.), находя- щихся в нижней части формы можно вводить или редактировать записи в таблице сотруд- ники.

Рисунок 10.7.Форма Сотрудники

Ввести в таблицу Сотрудники дополнительно шесть записей, одну удалить, две отредакти- ровать

Таблица 10.1.Кнопки управления и редактирования данных



Индивидуальные задания к практической работе Создать форму для таблицы Книги БД Библиотека

В диалоговом окне выбора полей для отображения выбрать все поля таблицы Книги В окне выбора стиля отображения полей и управляющих кнопок (см. рис.3.) в области

Style (стиль) выбрать стиль Embossed, а в области Buttontype (Типы кнопок управления) выбрать пункт NoButtons

Установить критерий упорядочения данных по полю kodknigi

В окне задания заголовка формы в области Typeatitleforyourform(Тип заголовка формы) указать имя формы Книги. Просмотреть вид создаваемой формы нажатием кнопки Preview (Просмотр).

После сохранения запустить форму Книги, она должна иметь вид как на рис.10.8.

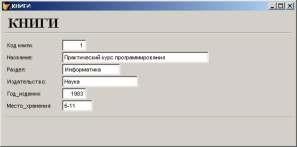


Рисунок 10.8. Форма Книги

Задание 2. Создание формы средствами конструктора форм

Запустить программу MicrosoftVisualFoxPro Открыть проект Информационная система

Открыть БД штат Выбрать в окне Project Manager вкладку Documents/Forms/New/New Form. Откроется окно FormDesigner (Конструктор форм) для создания новой формы Открыть окно окружения формы DataEnvironment (Среда окружения): пункт меню View/ DataEnvironment. Для размещения таблицы в среде окружения выбрать команду Add(Доба- вить) из меню DataEnvironment (Среда окружения) или вызвать контекстное меню на окне окружения формы DataEnvironmentи выбрать пункт Add. В появившемся диалоговом окне AddTableorView (Добавить таблицу или представление данных), выбрать из списка таблиц открытой базы штат таблицу Сотрудники, для которой создаѐтся форма, и нажать кнопку OK

Открыть окно свойств таблицы, размещѐнной в окне окруженияDataEnvironment. Для этого установить на таблицу Сотрудники курсор, вызвать контекстное меню правой клавишей мыши и выбрать команду Properties (Свойства)

В окне Properties (свойства) выделить свойство Order (Упорядочение). Для упорядочения данных в форме в поле коррекции свойства нажмите кнопку раскрытия списка и из списка индексов таблицы выберите индекс (id), по которому хотите упорядочить данные.

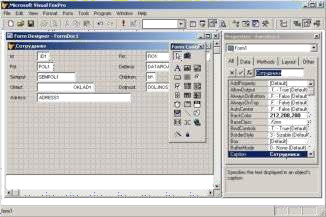
Закройте окно определения среды окружения DataEnvironment.

Для задания свойств формы установить курсор в форму, вызвать контекстное меню, вы- брать команду Properties (Свойства). Откроется окно Properties(Свойства). В его верхнем списке, указывающем название объекта, для которого осуществляется настройка свойств содержится текст Form1 (Форма1)

В окне Properties (Свойства) скорректировать свойство Caption (Надпись), введя в текстовое поле заголовок формыСотрудники тест

Свойство формы AutoCenter (Автоцентр) должно иметь значение True(Истина), чтобы форма располагалась в центре экрана

Изменить свойства FontName (Наименование шрифта), указав шрифт TimesNewRomanи FontSize (Размер шрифта) указав размер шрифта 12 После того как вы определили пара- метры формы, разместили в окружении используемые таблицы, можно приступать к разме- щению объектов в форме. Осуществим размещение полей таблицы Сотрудники и надписей к ним, используя FormBuilder (построитель формы)

Для запуска построителя форм установить курсор в форму, вызвать контекстное меню, вы- брать команду Builder (Построитель), откроется диалоговое окноFormBuilder (построитель формы), содержащее две вкладки: FieldSelection (Выбор поля) и Styles (Стиль). В вкладке FieldSelection (Выбор поля) из области Databasesandtables (Базы данных и таблицы) выбрать БД штат и таблицу Сотрудники, затем из списка Availablefields (Имеющиеся поля) перене- сти в Selectedfields(Выбранные поля) все необходимые поля используя кнопки расположен-

ные между списками. Перейти на вкладку Styles (Стиль), выбрать из списка стилей стиль Embossed и нажать кнопку OK. На форме будут размещены объекты формы (см. рис. 10.9.).



Рисунок 10.9. Окно FormDesigner(Конструктор форм) Рисунок 10.10. Объект - блок кно- пок управления

Для размещения на создаваемой форме кнопок управления воспользуемся ранее созданной формой Сотрудники. Открыть форму Сотрудники в режиме редактирования: окно программы ProjectManager/Documents/Forms/сотрудники/Modify. В появившемся окне FormDesigner формы Сотрудники нажатием левой клавиши мыши выделить объект BUTTONSET1 (блок кнопок управления, см. рис.10.10.) и нажать комбинацию клавиш Ctrl+Cна клавиатуре или установить курсор в форму, вызвать контекстное меню, выбрать команду Copy. Закрыть окно FormDesigner формы Сотрудники.

В окне FormDesigner формы Сотрудники тест установить курсор в форму, вызвать кон- текстное меню, выбрать команду Paste или нажать комбинацию клавиш Ctrl+V. Блок кно- пок управления записями таблицы (см. рис.10.10) в создаваемой форме будет скопирован. Задать цвет фона формы. Выделить в окнеProperties (Свойства) свойство формы BackColor (цвет фона), нажать расположенную с правой стороны поля редактирования свойства кнопку и в открывшемся диалоговом окне Цвет выберите цвет, который вы хотите исполь- зовать для фона

Расположить объекты формы в соответствии с рис.10.11. Для перемещения объектов формы можно использовать метод перетаскивания мышью или кнопками управления кур- сором на клавиатуре, предварительно выделив необходимый объект, а также комбинацией клавиш Ctrl+[кнопки управления курсором], использовать все выше перечисленные спо- собы, выяснить их различия. Для изменения размеров формы или объектов формы необхо- димо выделить объект (форму), подвести курсор к углу объекта (формы), когда курсор при- мет вид разнонаправленной стрелки нажать левую клавишу мыши и удерживая еѐ, изменить размеры объекта (формы)

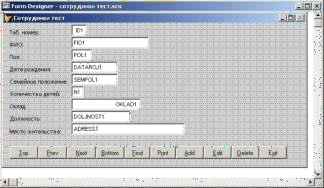


Рисунок 10.11 Вид отредактированной формы

Запустить отредактированную форму: установить курсор в форму, вызвать контекстное меню, выбрать команду RunForm или нажать кнопку [ ] на панели инструментов. Система попросить сохранить созданную форму, нажать кнопку Yes, в появившемся окне указать папку и имя файла (Сотрудники тест), сохранить.

Индивидуальные задания к практической работе Создать кнопки навигации в форме Книги

Открыть форму Книги в окне FormDesigner (Конструктор форм)

Нажать кнопку CommandGroup (Группа кнопок)  на панели инструментов FormControls (элементы управления формы) и щѐлкнуть в месте их предполагаемого размещения в форме

В Окне Properties (свойства) размещѐнного объекта выделить свойство ButtonCount (Коли- чество кнопок) и указать необходимое количество размещаемых в форме кнопок, указав число 5

Увеличить с помощью мыши размеры рамки, окружающей данный объект. Перейти в ре- жим редактирования: установить на объект курсор, вызвать контекстное меню и выбрать команду Edit (Редактировать). Выделяя поочерѐдно кнопки, переместить их, расположив горизонтально, в одну линию

Выйти из режима редактирования, щѐлкнув вне области объекта CommandGroup (Группа кнопок)

Скорректировать размер рамки, окружающей объект: в свойстве AutoSize (Авторазмер) установить значение True (Истина)

Открыть окно свойств объекта CommandGroup (Группа кнопок), нажать кнопку раскрытия списка в верхней части данного окна, поочерѐдно выбрать из списка элементы Command1, Command2, Command3, Command4, Command5 и, используя свойство

Caption(Надпись) задать названия кнопок соответственно: Первая, Следующая, Предыду- щая, Последняя, Выход (см. рис.10.12.)

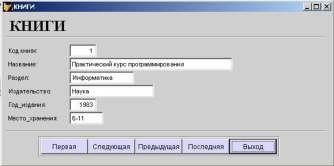


Рисунок 10.12. Форма Книги

Определить команды, которые будут выполняться при нажатии на созданные кнопки. Установить на объект CommandGroup (Группа кнопок) курсор, вызвать контекстное меню и выбрать команду Edit (Редактировать), выделить кнопку Первая, вызвать контекстное меню выбрать команду Code (Код программы), в появившемся окне ввести следующие ко- манды:

Для кнопки Первая:

Переходим на первую запись и обновляем информацию в форме IF !BOF() GO TOP ENDIF

\_screen.ActiveForm.Refresh()

После ввода кода программы закрыть окно.

Ввести аналогично команды для остальных кнопок: Для кнопки Следующая:

Переходим на следующую запись и обновляем информацию в форме IF !EOF() SKIP ENDIF

\_screen.ActiveForm.Refresh() Для кнопки Предыдущая:

Переходим на предыдущую запись и обновляем информацию в форме IF !BOF() SKIP - 1 ENDIF

\_screen.ActiveForm.Refresh() Для кнопки Последняя:

Переходим на последнюю запись и обновляем информацию в форме IF !BOF()

GO BOTTOM ENDIF

\_screen.ActiveForm.Refresh() Для кнопки Выход:

Запрашиваем и выходим, если Да

IFMESSAGEBOX("Выходить из формы?", 4+32+256, "Выход")=6

\_screen.ActiveForm.Release() ELSE

\_screen.ActiveForm.Refresh() ENDIF

После ввода команд закрыть окно процедур

Запустить форму на выполнение. Проверить результат проделанной работы

## Практическая работа №11 Добавление записей в табличный файл из двумер- ного массива. Работа с командами ввода-вывода. Использование функций для ра-

**боты с массивами**

**Цель работы:** овладение навыками изменения табличных файлов при помощи за- просов и функций

### Обработка запросов в VisualFoxPro

Одним из основных назначений разработанного приложения является быстрый по- иск информации в БД и получение ответов на разнообразные вопросы. Для этих целей в VisualFoxPro используются средства, называемые запросами.

При создании запросов в VisualFoxPro разработчик может использовать следующие средства:

* QueryWizard (Мастер запросов)
* QueryDesigner (Конструктор запросов)
* Команда SELECT языкаVisual FoxPro

Результатом запроса является таблица, которую можно сохранить в массиве, созда- ваемой новой таблице, отобразить на экране в режиме Browse (Просмотр) или вывести в виде отчѐта.

Для создания запросов можно использоватьQueryWizard (Мастер запросов), который последовательно запрашивает наименования таблиц, используемых в запросе, перечень по- лей таблиц, критерий упорядочения и условия фильтрации данных. С помощью QueryDesigner (Конструктор запросов) можно формировать различной сложности критерии для выбора записей из одной или нескольких таблиц, над полями, выбираемыми из таблиц с помощью запросов, можно выполнять различные вычисления.

В верхней части окна QueryDesigner (Конструктор запросов) расположена панель, на которой отображаются используемые в запросе таблицы. Ниже находятся вкладки, предна- значенные для выбора полей запроса и формирования условий выборки. Назначение этих вкладок приведено в таблице 1.. Открывая в окне QueryDesigner (Конструктор запросов) необходимые вкладки, можно выполнять следующие действия:

* Выбирать поля результирующей таблицы запроса
* Формировать вычисляемые поля
* Задавать критерии для выборки, группировки и упорядочения данных
* Указывать, куда выводить результаты выборки

Таблица 1. Назначение вкладок окна QueryDesigner (Конструктор запросов)



Таблица 2. Назначение команд меню Query и кнопок панели инструментов Query DesignerокнаQuery Designer (Конструкторзапросов)



**Задание для практической работы** Запустить программу MicrosoftVisualFoxPro Открыть проект Информационная система Открыть БД Штат проекта

Создать запрос о сотрудниках, имеющих более одного ребѐнка и получающих зар- плату менее 9000: в окне ProjectManager щѐлкнуть клавишей мыши на вкладке

Data/Queries/New/QueryWizard, в появившемся диалоговом окне WizardSelection (Выбор мастера) выбрать пункт QueryWizard (Мастер запросов) и нажать кнопку OK

В появившемся диалоговом окне выбора исходной таблицы и полей в области Databasesandtables (Базы данных и таблицы) выбрать необходимую БДштат иуказать таб- лицу Сотрудники. Из области Availablefields (Имеющиеся поля) выбрать поля Fio и Oklad. Для перехода к следующему шагу нажать кнопку Next(Далее)

В появившемся диалоговом окне выбора условий выборки в области Field(Поле) верхней строки выбрать поле Children (Количество детей) таблицы Сотрудники, в области Operator (Оператор Условия) выбрать условие morethen, в области Value (Выражение) уста- новить значение 1, в нижней строке внести следующие данные соответственно: в области Field(Поле) – поле Oklad (Зарплата), в области Operator (Оператор Условия) – lessthan, в области Value (Выражение) – 9000. Нажать кнопку Next(Далее)

В появившемся диалоговом окне выбора условия сортировки данных из области

Availablefields (Имеющиеся поля) выбрать поле Fio. Нажать кнопку Next(Далее)

Нажать кнопку Preview (Просмотр) и просмотреть результат запроса (запрос будет выглядеть в виде простой таблицы см. рис.11.1.), закрыть таблицу запроса. Нажать кнопку Finish (Готово), в появившемся диалоговом окне SaveAs (Сохранить как) указать необходи- мую папку и присвоить запросу имя Зарплата дети



Рисунок 11.1. Запрос Зарплата дети

Запустить запрос: в окне ProjectManager щѐлкнуть клавишей мыши на вкладке

Data/Queries/Зарплата дети/Run 10. Закрыть запрос Зарплата дети

Создать новый запрос, показывающий количество сотрудников организации, полу- чающих определѐнную зарплату: в окне ProjectManager щѐлкнуть клавишей мыши на вкладке Data/Queries/New/NewQuery. В появившемся диалоговом окне AddTableorView (Добавить таблицу или представление данных) в области Database (База данных) выбрать БД Штат, в области TablesinDatabaseтаблицу Сотрудники. Нажать кнопку Add, при этом в верхней области окна QueryDesigner (конструктор запросов) (см. рис.11.2.), находящегося позади окна AddTableorView (Добавить таблицу или представление данных) появится вы- бранная таблица. Нажать кнопку Close (Закрыть) для закрытия окна AddTableorView.

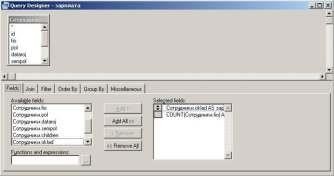


Рисунок 11.2. Окно QueryDesigner (конструктор запросов)

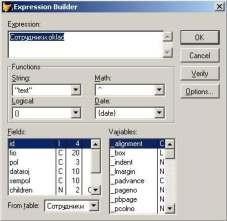
На вкладке Fields (Поля) окна QueryDesigner (конструктор запросов) в области Availablefields (Имеющиеся поля) выделить поле Сотрудники.oklad, оно автоматически пе- ренесѐтся в область Functionandexpressions (Функции и выражения), нажать расположен- ную справа от поля кнопку вызова построителя выражения, откроется диалоговое окно ExpressionBuilder (Построитель выражения) (см. рис.11.3.)

Рисунок 11.3. Диалоговое окно ExpressionBuilder (Построитель выражения)

В поле ввода Expression (Выражение) диалогового окна ExpressionBuilder (Построи- тель выражения) используя поля таблиц, расположенные в списке Fields (Поля), функции области Functions (Функции), и ввод данных с клавиатуры сформировать следующее выра- жение: Сотрудники.oklad AS Зарплата

Для проверки правильности набранного выражения нажать кнопку Verify (Про- верить),

нажать кнопку OK

Для переноса созданного выражения из области Functionandexpressions (Функции и выражения) в область SelectedFields (Выбранные поля) нажать находящуюся между дан- ными областями кнопку Add (Добавить)

Руководствуясь предыдущими пунктами 12-15 практической части создать следую- щее выражение:

COUNT(Сотрудники.fio) AS Количество\_сотрудников

Выбрать вкладку OrderBy (Упорядочение) окна QueryDesigner (конструктор запро- сов) и из области SelectedFields (Выбранные поля) в область Orderingcriteria (Критерий упорядочения) перенести поле Сотрудники.oklad

Выбрать вкладку GroupBy (Группировка) окна QueryDesigner (конструктор запро- сов) и из области Availablefields (Имеющиеся поля) в область GroupedFields (Поля группи- ровки) перенести поле Сотрудники.oklad

Выбрать пункт меню Query (Запрос) окна программы VisualFoxPro, выбрать команду Comments (Комментарии) (см. таблицу 2.). В появившемся окне Comment(Комментарий) в поле ввода области AddComment (Добавить комментарий) ввести строку: Запрос, показы- вающий количество сотрудников организации, получающих определѐнную зарплату. За- крыть окно Comment(Комментарий)

Запустить созданный запрос: установить курсор в окне QueryDesigner (конструктор запросов), вызвать контекстное меню, выбрать команду RunQuery, или нажать кнопку [ ] на панели инструментов. Закрыть запрос

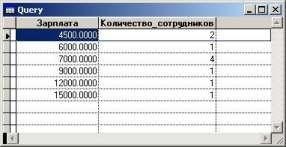


Рисунок 11.4. Запрос Зарплата

Просмотреть SQL-оператор, соответствующий запросу: установить курсор в окне QueryDesigner (конструктор запросов), вызвать контекстное меню, выбрать команду ViewSQL (Показать SQL), откроется диалоговое окно, в котором отображается SQL-опера- тор, соответствующий данному запросу. Закрыть окно QueryDesigner (конструктор запро- сов), система предложит сохранить запрос, нажать кнопку Yes (Да), в появившемся диало- говом окне SaveAs (Сохранить как) указать необходимую папку и присвоить запросу имя Зарплата

# Индивидуальные задания к практической работе

Создать многотабличный запрос, выводящий в таблицу названия, год издания и ав- тора книг по информатике

Создать новый запрос, добавить таблицы Авторы, Автор\_книги и Книги из БД Биб- лиотека в окно QueryDesigner (конструктор запросов)

Из области Availablefields (Имеющиеся поля) вкладки Fields (Поля) внести в область SelectedFields (Выбранные поля) используя построитель выражения ExpressionBuilder сле- дующие выражения:

Книги.nazvanie AS название\_книги Книги.razdel AS раздел Книги.godizdan AS год\_изда- ния

ALLTRIM(Авторы.familiya)+" "+ALLTRIM(Авторы.imya) AS автор

Необходимо задать условие выбора записей из таблицы (Книги только по информа- тике) по полю razdel (Раздел). Перейти на вкладку Filter (Фильтр), в области FieldName (Имя поля) нажать кнопку раскрытия списка и выбрать поле Книги.razdel, в области Criteria (Кри- терии) выбрать значение ==, в области Example (Образец) ввести значение ― Информатика‖ Запустить запрос и посмотреть результат (см. рис.11.5.), закрыть запрос, закрыть ок- ноQueryDesigner (конструктор запросов), сохранить запрос, присвоив ему имя Книги по ин-

форматике

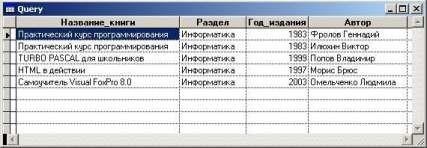


Рисунок 11.5. Запрос Книги по информатике

## Практическая работа №12 Создание меню различных видов. Модификация и

**управление меню**

**Цель работы:** овладение навыками создания и модификации структуры меню

# Создание меню в VisualFoxPro

В соответствии со стандартами Windows в любом приложении рекомендуется иметь строку меню, которая в VisualFoxPro содержит команды, предназначенные для вызова форм, формирования отчѐтов, запросов и т.д.

Строкой меню называется горизонтальное меню, располагаемое в верхней части экрана. Примером строки меню является основное меню VisualFoxPro, а также меню про- грамм, работающих в среде Windows. Созданное в конструкторе MenuDesigner (Конструк- тор меню) меню может замещать основное меню VisualFoxPro или добавляться к нему.

Для создания меню необходимо выполнить следующие действия:

* Открыть окно конструктора меню MenuDesigner (Конструктор меню)
* Описать вид меню, текст, пункты меню и его атрибуты
* Сгенерировать меню. При этом создаѐтся программа, которая в результате запускается на выполне- ние

VisualFoxPro имеется возможность создания меню в виде строки Menu, или всплы- вающего меню Shortcut, в котором основные пункты расположены по вертикали.

Таблица 12.1.Назначение кнопок конструктора меню MenuDesigner (Конструктор меню)



Таблица 12.2. Типы меню



Таблица 12.3. Типы пункта меню



**Задание для практической работы** Запустить программу MicrosoftVisualFoxPro Открыть проект Информационная система

Открыть окно MenuDesigner (Конструктор меню): в окне ProjectManager щѐлкнуть клавишей мыши на вкладке Other/Menus/New, в появившемся диалоговом окне NewMenu (Новое меню) выбрать пункт Menu(Меню), появится окно MenuDesigner (Конструктор меню) (см. рис.12.1.)

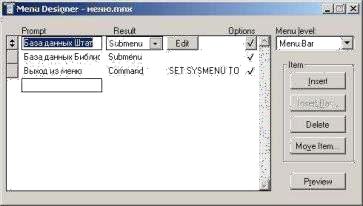


Рисунок 12.1. Окно MenuDesigner (Конструктор меню)

Область конструктора меню, над которой расположены надписи Prompt (Приглаше- ние), Result (Результат) и Options(Опции), предназначена для формирования меню. В поле Prompt (Приглашение) вводят наименования пунктов меню, раскрывающийся список Result (Результат) используется для указания типа пункта меню, а кнопка Options (Опции) откры- вает диалоговое окно PromptOptions (Опции элемента меню), в котором можно определить дополнительные параметры данного элемента меню. В списке Menulevel (Уровень меню) указывается уровень текущего меню. Слева в конструкторе меню размещены кнопки (см. таблицу 12.1.)

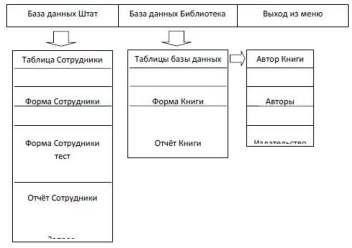


Рисунок 12.2. Схема создаваемого меню Меню

В поле Prompt (Приглашение) ввести наименование первого пункта меню – База дан- ных Штат, нажать для перехода на следующее поле клавишу Enter или Tab на клавиатуре, курсор окажется в списке Result (Результат) (см. рис.1.) Указать тип пункта меню Submenu (Подменю), нажать кнопку Options(Опции), откроется диалоговое окно PromptOptions (Опции элемента меню), в котором, в области Shortcut (Всплывающее меню) установить курсор в поле KeyLabel (Метка) и нажать комбинацию клавиш Alt + F1 на кла- виатуре. Мы определили комбинацию клавиш быстрого вызова пункта меню База данных Штат. Нажать кнопку OK

Перейти на следующую строку и ввести наименование второго пункта – База данных Библиотека, тип пункта меню Submenu (Подменю), определить комбинацию клавиш Alt + F2 быстрого вызова пункта меню База данных Библиотека (см. пункт 4. лабораторной ра- боты)

Перейти на следующую строку и ввести наименование третьего пункта – Выход из меню, в списке Result (Результат) указать тип пункта меню Command (Команда), в появив- шейся справа строке ввода ввести команду – SET SYSMENU TO DEFAULT. Определить комбинацию клавиш Alt + F4 быстрого вызова действия пункта меню Выход из меню

Пункты меню База данных Библиотека и База данных Штат имеют свои подменю. Для создания подменю перейти на строку База данных Библиотека, нажать кнопку Create (Создать), находящуюся справа от списка Result (Результат), на экране появится пустое окно конструктора меню. Ввести в поле Prompt (Приглашение) наименование подпункта – Таблицы базы данных, в списке Result (Результат) указать тип пункта меню Submenu (Под- меню), нажать кнопку Create (Создать), так как он тоже имеет подменю (см.рис.2.)

В появившемся пустом окне конструктора меню ввести в поле Prompt(Приглашение) наименование подпункта – Автор книги, в списке Result (Результат) указать тип пункта меню Procedure (Процедура), нажать кнопку Create (Создать) и в открывшемся окне редак- тирования процедуры ввести следующие команды:

OPEN DATABASE \<Имя вашей папки>\библиотека.dbc USE библиотека!автор\_книги SELECTавтор\_книги BROWSE

CLOSE DATABASES

Закрыть окно редактирования процедуры, перейти на следующую строку и ввести наименование следующего пункта – Авторы, в списке Result (Результат) указать тип пункта меню Procedure (Процедура), нажать кнопку Create (Создать) и в открывшемся окне редактирования процедуры ввести следующие команды:

OPEN DATABASE \<Имя вашей папки>\библиотека.dbc USE библиотека!авторы SELECT авторы

BROWSE

CLOSE DATABASES

Закрыть окно редактирования процедуры, перейти на следующую строку и ввести наименование следующего пункта – Издательство, в списке Result (Результат) указать тип пункта меню Procedure (Процедура), нажать кнопку Create (Создать) и в открывшемся окне редактирования процедуры ввести следующие команды:

OPEN DATABASE \<Имя вашей папки>\библиотека.dbc USE библиотека!издательство SELECT издательство

BROWSE

CLOSE DATABASES

Закрыть окно редактирования процедуры, перейти на следующую строку и ввести наименование следующего пункта – Книги, в списке Result (Результат) указать тип пункта меню Procedure (Процедура), нажать кнопку Create (Создать) и в открывшемся окне редак- тирования процедуры ввести следующие команды:

OPEN DATABASE \<Имя вашей папки>\библиотека.dbc USE библиотека!книги SELECT книги

BROWSE

CLOSE DATABASES

Закрыть окно редактирования процедуры, перейти на следующую строку и ввести наименование следующего пункта – Разделы, в списке Result (Результат) указать тип пункта меню Procedure (Процедура), нажать кнопку Create (Создать) и в открывшемся окне редак- тирования процедуры ввести следующие команды:

OPEN DATABASE \<Имя вашей папки>\библиотека.dbc USE библиотека!разделы SELECT разделы

BROWSE

CLOSE DATABASES

Закрыть окно редактирования процедуры. Для перехода в меню верхнего уровня ис- пользовать список MenuLevel (Уровень меню): выбрать из выпадающего списка MenuLevel (Уровень меню) пункт – Базаданных

Под строкой Таблицы базы данных ввести наименование следующего пункта – Форма Книги, в списке Result (Результат) указать тип пункта меню Command(Команда), в появившейся справа строке ввода ввести команду – DO FORM \< Имя вашей папки>\книги.scx

Перейти на следующую строку и ввести наименование следующего пункта – Отчѐт Книги, в списке Result (Результат) указать тип пункта меню Command(Команда), в появив- шейся справа строке ввода ввести команду – REPORT FORM \< Имя вашей папки>\книги.frx PREVIEW

Перейти на следующую строку и ввести наименование следующего пункта – Запрос, выводящий на экран названия, год издания и авторов книг по информатике, в списке Result (Результат) указать тип пункта менюProcedure (Процедура), нажать кнопку Create (Создать) и в открывшемся окне редактирования процедуры ввести следующие команды:

OPEN DATABASE \<Имя вашей папки>\библиотека.dbc DO "\<Имя вашей папки>\книги по информатике.qpr" CLOSE DATABASES

Закрыть окно редактирования процедуры. Для просмотра создаваемых пунктов меню, нажать кнопку Preview (Просмотр), при этом основное меню VisualFoxPro будет за- менено создаваемым меню. После просмотра нажать кнопку OKдиалового окна Preview (Просмотр)

Для улучшения внешнего вида, а также для объединения в группы схожих по смыслу команд, в меню можно использовать разделительные линии. Разделительные линии пред-

ставляют собой пункт меню, в котором в поле ввода Prompt(Приглашение) вместо наиме- нования пункта вводятся символы \-. Также VisualFoxPro позволяет справа от команд поль- зовательскогоменю расположить графическое изображение, для этого необходимо выпол- нить следующие действия: нажать кнопку Options(Опции) для необходимого пункта меню, откроется диалоговое окно PromptOptions (Опции элемента меню), в котором, в области Picture(Изображение) указать опциюResource (Ресурс), нажать кнопку, располагаемую справа от поля, находящегося под опцией. Откроется диалоговое окно InsertSystemMenuBar (Вставить из системного меню). Из списка графических изображений, используемых VisualFoxPro в системном меню выбрать наиболее соответствующее пункту меню значение и нажать кнопку OK. Выбранное значение переносится в область просмотра области Picture(Изображение). Нажать кнопку OK для закрытия окна PromptOptions (Опции эле- мента меню)

Для перехода в меню верхнего уровня выбрать из выпадающего списка MenuLevel (Уровень меню) пункт – MenuBar

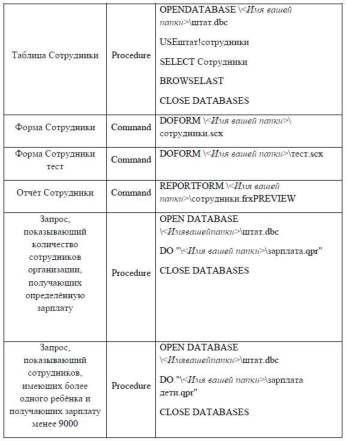
Указать месторасположение создавае-

мого меню: пункт меню VisualFoxProView/GeneralOptions, откроется диалоговое окно GeneralOptions (Основные параметры), в области Location (Размещение) выбрать пункт Append (Добавить), для добав- ления создаваемого меню в основное меню VisualFoxPro

# Индивидуальные задания к практической работе

Добавить в пункт меню – База данных Штат подпункты, определить действия для этих подпунктов меню в соответствии с таблицей 12.4.

Таблица 12.4. Наименование пунктов меню и определение их действий



Сохранить созданное меню: пункт меню File/SaveAs, в появившемся диалоговом окне указать имя своей папки, куда следует сохранить файл, присвоить имя Меню и нажать кнопку Save (Сохранить)

Необходимо сгенерировать меню: пункт меню Menu/Generate, откроется диалоговое

окно

GenerateMenu (Генерация меню), нажать кнопку Generate (Генерация)

После завершения генерации запустить программу меню на выполнение: в окне про-

екта ProjectManagerустановить курсор на наименование созданного меню Меню и нажать кнопку Run (Запустить)

Практическая работа №14 «Создание рабочих и системных окон. Добавление эле- ментов управления рабочим окном»

Цель работы: овладение практическими навыками работы с системными окнами, со- здание отчетов по представленным данным

Отчеты в VisualFoxPro

Отчет – форматированное представление данных, выводимое на экран, принтер или в файл. Отчет, создаваемый в VisualFoxPro, может быть представлен в табличном виде или в свободной форме. Табличные отчеты используются для печати данных, представленных

в виде списка. При создании отчета в VisualFoxProразработчикможет использовать следу- ющие средства:

ReportWizard – Мастер отчета. Позволяет быстро создать отчѐт, применяя сорти- ровку, группировку данных и заданный разработчиком стиль оформления;

ReportDesigner – Конструктор отчета. Позволяет разрабатывать собственные отчѐты или модифицировать отчѐты, созданные с помощью мастера;

QuickReport – Быстрый отчет. Данное средство предназначено для размещения в конструкторе отчѐта полей и задания среды окружения.

Задание для практической работы Запустить программу MicrosoftVisualFoxPro От- крыть проект Информационная система

Открыть БД проекта. Для этого необходимо на вкладке Data (Данные) установить курсор на еѐ название и нажать кнопку Open (Открыть) окна проекта. При этом на стандарт- ной панели инструментов в списке Databases (Базы данных) появится название открытой БД

Перейти на вкладку Documents/Reports/New/New Report /Report Wizard

В появившемся диалоговом окне WizardSelection указать тип создаваемого отчета ReportWizard(Мастер отчетов) и нажать кнопку OK

Появляется первое диалоговое окно мастера отчѐта (см. рис.13.1.). Выбрать из верх- него списка области Databasesandtables (Базы данных и таблицы) базу данных штат, а из нижнего – таблицу сотрудники

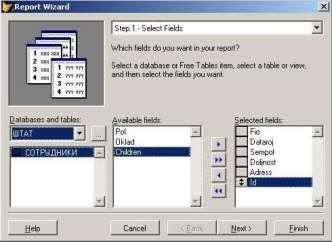


Рисунок 13.1. Диалоговое окно создания однотабличного отчета с помощью мастера. Перенести из списка Availablefields (Имеющиеся поля) в список Selectedfields (Вы- бранные поля) поля, которые вы хотите разместить в создаваемом отчете (в соответствии с рис.13.1.). После формирования списка отображаемых в отчете полей нажать кнопку Next

(Далее) для перехода к следующему шагу в создании отчета

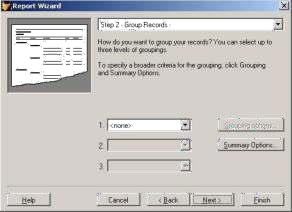
В следующем окне мастера создания отчета указать поля, по которым будут осу- ществляться группировка данных в отчете (рис.13.2.)

Рисунок 13.2. Диалоговое окно определения полей для группировки данных в отчете.

В центре диалогового окна расположены три раскрывающихся списка, позволяющих задать до трех группировок данных в отчете. Эти списки содержат все поля таблицы. Для осуществления группировки данных в отчете выбрать нужное поле из раскрывающегося списка 1. При создании второй и третьей группировки используются, соответственно, списки 2 и 3. В нашем примере группировка данных по полям не осуществляется, это диа- логовое окно мастера отчѐтов необходимо пропустить и нажать кнопку Next

В следующем диалоговом окне мастера задается стиль отображения объектов в от- чете (рис.13.3.). Список Style (Стиль) содержит несколько вариантов отображения объектов (полей, линий, заголовок и т.д.) в отчете. С помощью области просмотра в верхнем левом углу диалогового окна можно просмотреть тот стиль, который мы выбрали. Выбрав стиль, нажать кнопку

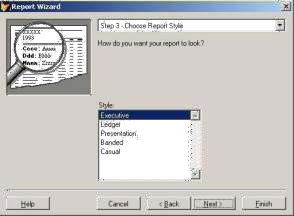


Рисунок 13.3. Диалоговое окно выбора стиля отображения объектов.

Указать порядок размещения объектов в отчете (рис.13.4.) и ориентацию страницы отчета. В области NumberofColumns (Количество колонок) указать 1, в области Orientation (Ориентация) Указать Portrait (Книжная), в области FieldLayout (Расположение полей) ука- зать Rows (Строка). После установки требуемых опций, нажать кнопку Next

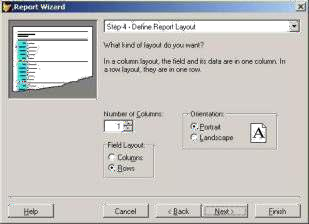


Рисунок 13.4. Диалоговое окно установки порядка размещения объектов.

Задать поле, по которому требуется упорядочение данных в отчете в соответствии с рис.13.5. Из списка Availablefieldsorindextag (Выбранные поля и индексы) перенести в список Selectedfields (Выбранные поля) поле Fio (упорядочивание данных по фами- лии). Для переноса полей используйте кнопку Add (Добавить). Сформировав список полей, нажать кнопку Next (Далее);

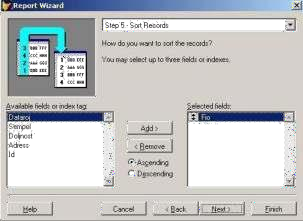


Рисунок 13.5. Диалоговое окно установки критерия упорядочения данных.

На заключительном шаге создания отчета в области Typeatitleforyourreport (Вве- дите имя заголовка отчета) указать имя отчѐта Сотрудники. Воспользовавшись кнопкой Preview (Просмотр) просмотреть, как будет выглядеть создаваемый отчет, после просмотра нажать кнопку ClosePreviewна панелиPrintPreview(см. рис.13.6.),если что-то не так, верни- тесь к предыдущим шагам, воспользовавшись кнопкой Back. Нажать кнопку Finish (Готово) и в появившемся диалоговом окне SaveAs (Сохранить как) указать папку, в которой будет размещен отчет Сотрудники.



Рисунок 13.6. Панель просмотра отчѐтаPrintPreview

Просмотреть созданный отчѐт: в окне ProjectManager выбрать вкладку Documents/Reports/Сотрудники/Preview или команду Preview (Просмотр) из меню про- граммы View (Вид)

Открыть отчѐт Сотрудники в окне ReportDesigner (конструктор отчѐта): в окне ProjectManager выбрать вкладку Documents/Reports/Сотрудники/Modify

Таблица 13.1. Типы полос отчѐта

Выделить все объекты в полосе Title (Титул) и перенести их полосу PageHeader (Верхний колонтитул) используя метод перетаскивания мышью или кнопками управления курсором на клавиатуре, предварительно выделив необходимый объект, или комбинацией клавиш Ctrl+[кнопки управления курсором]

В полосе Title (Титул) напечатать следующий текст: ― Отчѐт - форматированное представление данных, выводимое на экран, принтер или в файл. Табличный отчѐт исполь- зуется для печати данных, представленных в виде списка‖

Поместить набранный текст в прямоугольник со скругленными краями: нажать на панели инструментов ReportControls кнопку  поместить данный объект в отчѐт. От- редактировать его расположение в соответствии с рис.13.7.

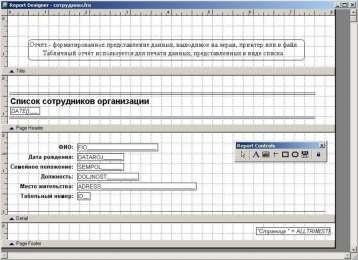


Рисунок 13.7. Окно ReportDesigner (Конструктор отчѐтов)

В процессе работы просматривайте результаты редактирования, используя пункт меню View/Preview

В полосе Detail (Детали), ниже поля Табельный номер провести линию, используя кнопку  панели инструментов ReportControls и отредактировать еѐ расположение в от- чѐте в соответствии с рис.7.

В полосе PageFooter (Нижний колонтитул) переместить элемент  в правый нижний угол в соответствии с рис.137.

Вызвать свойства элемента  отчѐта двойным щелчком мыши или через контекстное меню команда Properties и в появившемся диалоговом окне ReportExpression (Выражение отчѐта) в поле ввода Expression заменить словоPage на слово Страница, нажать OK

Просмотреть результат выполненной работы: пункт меню View/Preview

# Индивидуальные задания к практической работе

Создать отчет для таблицы Книги БД Библиотека

В диалоговом окне ReportWizard(Мастер отчетов) (см. рис.1.) выбрать следующие поля таблицы Книги: Nazvanie, Razdel, Izdat, Godizdan

В диалоговом окне выбора стиля отображения объектов (см. рис.3.) в поле области Style (Стиль) выбрать стиль Executive

В диалоговом окне установки порядка размещения объектов (см. рис.4.) в поле об- ластиNumberofColumns (Количество колонок) указать 1, в области FieldLayout (Расположе- ние полей) указать Rows (Строка)

В диалоговом окне установки критерия упорядочения данных (см. рис.5.) из списка Availablefieldsorindextag (Выбранные поля и индексы) перенести в список Selectedfields (Выбранные поля) поле Nazvanie

Просмотреть вид создаваемого отчета нажатием кнопки Preview (Просмотр). После сохранения запустить отчет Книги, она должна иметь вид как на рис.13.8.

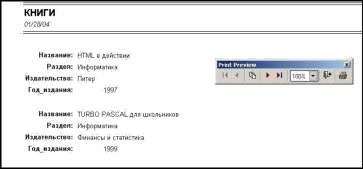


Рисунок 13.8. Отчѐт Книги

## Практическая работа №13 Создание файла проекта базы данных. Создание интерфейса входной формы. Использование исполняемого файла проекта БД, при-

**емы создания и управления**

**Цель работы:** овладеть навыками проектирования предметной области, создания интерфейсов и применения исполняемого файла БД

# Проектирование базы данных с использованием ER-технологии

Для заданной предметной области должен быть определен состав реляционных таб- лиц и логические связи между таблицами. Для каждого атрибута должны быть заданы тип и размер данных, ограничения целостности. Для каждой таблицы – первичный ключ, по- тенциальные ключи и внешние ключи.

Разработка логической модели методом «сущность-связь» (ER-методом) предусмат- ривает выполнение следующих шагов, детально описанных в работе:

1. построение ER-диаграммы, включающей все сущности и связи, важные с точки зрения интересов предметной области;
2. анализ связей и определение их характеристик – степени связи, мощности и класса принад- лежности;
3. построение набора предварительных отношений с указанием предполагаемого первичного ключа для каждого отношения;
4. подготовка списка всех неключевых атрибутов и назначение каждого из этих атрибутов одному из предварительных отношений;
5. проверка нахождения всех полученных отношений в нормальной форме Бойса-Кодда;
6. построение модели данных.

# Создание и связывание таблиц базы данных в среде MySQL

Рассмотрим следующие вопросы:

* + создание и выбор базы данных;
  + создание таблиц;
  + столбцы и типы данных в MySQL;
  + создание индексов;
  + удаление таблиц, индексов и баз данных;
  + изменение структуры таблиц.

Базы данных, таблицы и индексы легко создаются в рамках графического интер- фейса MySQL, но мы будем использовать монитор MySQL (клиент командной строки), чтобы лучше понять структуру БД, таблиц и индексов.

# Чувствительность к регистру и идентификаторы.

* + Имена БД подчиняются тем же правилам зависимости от регистра символов, каким следуют ката- логи операционной системы. Имена таблиц следуют тем же правилам, что и имена файлов. Все остальное не зависит от регистра.
  + Все идентификаторы, кроме имен псевдонимов, могут содержать до 64 символов. Имена псевдони- мов могут иметь до 255 символов.
  + Идентификаторы могут содержать любые допустимые символы, но имена баз данных не могут со- держать символы /, \ и . , а имена таблиц – символы . и /.
  + Зарезервированные слова можно использовать для идентификаторов, если заключить их в кавычки.

**Комментарий в SQL**. Начинается с двух дефисов (--), за которыми должен следо- вать пробел. Кроме того, MySQL содержит ряд собственных комментариев. Shell-коммен- тарий # действует аналогично – все, что расположено правее его, является текстом коммен- тария. С- комментарий /\* \*/ является многострочным – комментарий начинается с /\* и за- канчивается, когда встретится завершение \*/.

**Создание и выбор базы данных.** Осуществляется с помощью оператора

*CREATE DATABASE имя\_базы\_данных;*

Убедиться в том, что оператор выполнил задачу, можно с помощью оператора

*SHOW DATABASES;*

Теперь имеется пустая БД, ожидающая создания таблиц. Прежде чем работать с БД, необходимо выбрать эту БД с помощью оператора

*USE имя\_базы\_данных;*

Теперь все действия по умолчанию будут применяться именно к этой БД.

**Создание таблиц.** Используется оператор *CREATE TABLE*, который в общем виде выглядит следующим образом:

*CREATE [TEMPORARY] TABLE [IF NOT EXISTS]*

*имя\_таблицы (определение таблицы) [TYPE=тип\_таблицы];*

Ключевое слово *TEMPORARY* используется для создания таблиц, которые будут су- ществовать только в текущем сеансе работы с БД и будут автоматически удалены, когда сеанс завершится.

При использовании выражения *IF NOT EXISTS* таблица будет создана только в том случае, если еще нет таблицы с указанным именем.

Создать таблицу с такой же схемой, как у существующей, позволяет команда

*CREATE [TEMPORARY] TABLE [IF NOT EXISTS]*

*имя\_таблицы LIKE имя\_старой\_таблицы;*

После имени таблицы в скобках объявляются имена столбцов, их типы и другая ин- формация. В определение столбца можно добавить следующие описания.

* + Объявить для любого столбца *NOT NULL* или *NULL* (столбцу запрещено или не запрещено содер- жать значения *NULL*). По умолчанию – *NULL*.
  + Объявить для столбца значение по умолчанию, используя ключевое слово *DEFAULT*,

за которым должно следовать значение по умолчанию.

* + Использовать ключевое слово *AUTO\_INCREMENT*, чтобы генерировать порядковый номер. Авто- матически генерируемое значение будет на единицу большим, чем наибольшее значение в таблице. Первая введенная строка будет иметь порядковый номер 1. В таблице можно иметь не более одного столбца *AUTO\_INCREMENT*, и он должен индексироваться.
  + Объявить столбец первичным ключом таблицы с помощью выражения *PRIMARY KEY*.
  + Объявить столбец внешним ключом, используя выражение *FOREIGN KEY*, с ссылкой на соответ- ствующую таблицу с помощью выражения *REFERENCES*.
  + Индексировать столбец с помощью слов *INDEX* или *KEY* (синонимы). Такие столбцы не обяза- тельно должны содержать уникальные значения.
  + Индексировать столбец с помощью слова *UNIQUE,* которое используется для указания того, что столбец должен содержать уникальные значения.
  + Создать полнотекстовые индексы на основе столбцов типа *TEXT, CHAR* или *VARCHAR*

с помощью слова *FULLTEXT* (только с таблицами *MyISAM*).

После закрывающей скобки можно указать тип таблицы:

* + *MyISAM* – таблицы этого типа являются «родными» для MySQL, работают очень быстро и поддер- живают полнотекстовую индексацию;
  + *InnoDB* – ACID-совместимый механизм хранения, поддерживающий транзакции, внешние ключи, каскадное удаление и блокировки на уровне строк;
  + *BDB (Berkeley DB)* – является механизмом хранения, который обеспечивает поддержку транзакций и блокировки на уровне страниц;
  + *MEMORY (HEAP)* – таблицы целиком хранятся в оперативной памяти и никогда не записываются на диск, поэтому работают очень быстро, но ограничены в размерах и не допускают возможности восстановления в случае отказа системы;
  + *MERGE* – тип позволяет объединить несколько таблиц *MyISAM* с одной структурой, чтобы к ним можно было направлять запросы как к одной таблице;
  + *NDB Cluster* – тип предназначен для организации кластеров MySQL, когда таблицы распределены между несколькими компьютерами, объединенными в сеть;
  + *ARCHIVE* – тип введен для хранения большого объема данных в сжатом формате; таблицы поддер- живают только два SQL-оператора: *INSERT* и *SELECT,* причем оператор *SELECT* выполняется по методу полного сканирования таблицы;
  + *CSV* – формат представляет собой обычный текстовый фал, записи в котором хранятся в строках, а поля разделены точкой с запятой (широко распространен в компьютерном мире, любая программа, поддерживающая CSV-формат, может открыть такой файл);
  + *FEDERATED* – тип позволяет хранить данные в таблицах на другой машине сети (при создании

таблицы в локальной директории создается только файл определения структуры таблицы, а все дан- ные хранятся на удаленной машине).

MySQL поддерживает следующие типы данных, допустимые для столбцов:

* + числовые;
  + строковые;
  + календарные;
    - *NULL* – специальный тип, обозначающий отсутствие информации. Числовые типы используются для хранения чисел и представляют два подтипа:
  + точные числовые типы;
  + приближенные числовые типы.

К точным числовым типам (табл. 1) относятся целый тип *INTEGER* и его вариации, а также вещественный тип *DECIMAL* (синонимы *NUMERIC* и *DEC*). Последний использу- ется для представления денежных данных.

Числовые типы могут характеризоваться максимальной длиной *М*. Для типа *DECIMAL* параметр *М* задает число символов для отображения всего числа, a *D* – для его дробной части. Например: *b\_price DECIMAL (5, 2).* Цифра 5 определяет общее число сим- волов под число, а цифра

2 – количество знаков после запятой (интервал величин от –99.99 до 99.99). Можно не ис- пользовать параметры вообще, указать только общую длину или указать длину и число де- сятичных разрядов.

Объявления точных числовых типов можно завершать ключевыми словами *UNSIGNED* и (или) *ZEROFILL*. Ключевое слово *UNSIGNED* указывает, что столбец содер- жит только положи- тельные числа или нули. Ключевое слово *ZEROFILL* означает, что число будет отображаться с ведущими нулями.

Таблица 14.1. Числовые типы данных

|  |  |  |
| --- | --- | --- |
| **Тип** | **Объем памяти** | **Диапазон** |
| *TINYINT (M) TINYINT UN- SIGNED* | 1 байт | от -128 до 127 (от -27 до 27-1)  от 0 до 255 (от 0 до 28-1) |
| *SMALLINT (M) SMALLINT UNSIGNED* | 2 байта | от -32 768 до 32 767 (от -215 до 215-1)  от 0 до 65 535 (от 0 до 216-1) |
| *MEDIUMINT (M) MEDI- UMINT UNSIGNED* | 3 байта | от -8 388 608 до 8 388 607 (от -223 до 223-1)  от 0 до 16 777 215 (от 0 до 224-1) |
| *INT (INTEGER) (M) INT UNSIGNED* | 4 байта | от -2 147 683 648 до 2 147 683 647 (от -231 до 231-1)  от 0 до 4 294 967 295 (от 0 до 232-1) |
| *BIGINT (M) BIGINT UN- SIGNED* | 8 байт | (от-263 до263-1)  (от 0 до 264 -1) |
| *BIT (M)* | (*М*+7)/8 байт | От 1 до 64 битов, в зависимости от значения *М* |
| *BOOL, BOOLEAN* | 1 байт | 0 (*false*) либо 1 (*true*) |
| *DECIMAL (M, D),*  *NUMERUC (M, D)* | *М* + 2 байта | Повышенная точность, зависит от параметров  *М* и *D* |

К приближенным числовым типам (табл. 14.2) относятся:

* + *FLOAT* – представление чисел с плавающей запятой с обычной точностью;
    - *DOUBLE* – представление чисел с плавающей запятой с двойной точностью. Таб- лица 14.2. Приближенные к числовым типы данных

|  |  |  |
| --- | --- | --- |
| **Тип** | **Объем памяти** | **Диапазон** |
| *FLOAT (М, D)* | 4 байта | Минимальное по модулю значение 1.175494351\*10-39  Максимальное по модулю значение 3.402823466\*1038 |
| *DOUBLE (M, D),*  *REAL (M,D),*  *DOUBLE PRECISION (M,D)* | 8 байт | Минимальное по модулю значение 2.2250738585072014\*10-308  Максимальное по модулю значение 1.797693134862315\*10308 |

Числовые типы с плавающей точкой также могут иметь параметр *UNSIGNED*. Атри- бут предотвращает хранение в столбце отрицательных величин, но максимальный интервал величин столбца остается прежним.

Приближенные числовые данные могут задаваться в обычной форме (например, 45.67) и в форме с плавающей точкой (например, 5.456Е-02 или 4.674Е+04).

Текстовые типы и строки (табл. 3):

* + *CHAR* – хранение строк фиксированной длины;
  + *VARCHAR* – хранение строк переменной длины;
  + *TEXT, BLOB* и их вариации – хранение больших фрагментов текста;
    - *ENUM* и *SET –* хранение значений из заданного списка. Таблица 14.3. Тестовые типы и строки

|  |  |  |
| --- | --- | --- |
| **Тип** | **Объем памяти** | **Максимальный размер** |
| *CHAR(M)* | *М* символов | *М* символов |
| *VARCHAR(M)* | *L*+1 символов | *М* символов |
| *TINYBLOB, TINYTEXT* | *L*+1 символов | 28-1 символов |
| *BLOB, TEXT* | *L*+2 символов | 216-1 символов |
| *MEDIUMBLOB, MEDIUMTEXT* | *L*+3 символов | 224-1 символов |
| *LONGBLOB,*  *LONGTEXT* | *L*+4 символов | 232-1 символов |
| *ENUM(*'*value 1*'*,* '*value2* '*, ...)* | 1 или 2 байта | 65 535 элементов |
| *SET(*'*value 1*'*,* '*value2*'*, ...)* | 1, 2, 3, 4 или 8 байт | 64 элемента |

Здесь *L* – длина хранимой в ячейке строки, а приплюсованные к *L* байты – накладные расходы для хранения длины строки.

Для строк *VARCHAR* требуется количество символов, равное длине строки плюс 1 байт, тогда как тип *CHAR(M)*, независимо от длины строки, использует для ее хранения все *М* символов. Тип *CHAR* обрабатывается эффективнее переменных типов. Нельзя смешивать в таблице столбцы *CHAR* и *VARCHAR*. Если есть столбец переменной длины, все столбцы типа *CHAR* будут приведены к типу *VARCHAR*.

Типы *BLOB* и *TEXT* аналогичны и отличаются в деталях. При выполнении операций над столбцами типа *TEXT* учитывается кодировка, а типа *BLOB* – нет. Тип *TEXT* использу- ется для хранения больших объемов текста, тип *BLOB* – для больших двоичных объектов (электронные документы, изображения, звук). Основное отличие *TEXT* от *CHAR* и *VARCHAR* – поддержка полнотекстового поиска.

Строки типов данных *ENUM* и *SET* принимают значения из заданного списка. Зна- чение типа *ENUM* должно содержать точно одно значение из указанного множества, тогда как столбцы *SET* могут содержать любой или все элементы заданного множества одновре- менно. Для типа *SET* (как и для *ENUM*) при объявлении задается список возможных значе- ний, но ячейка может принимать любое значение из списка, а пустая строка означает, что ни один из элементов списка не выбран.

Типы *ENUM* и *SET* задаются списком строк, но во внутреннем представлении эле- менты множеств сохраняются в виде чисел. Элементы типа *ENUM* нумеруются последова- тельно, начиная с 1. Под столбец может отводиться 1 байт (до 256 элементов в списке) или 2 байта (от 257 до 65536 элементов в списке). Элементы типа *SET* обрабатываются как биты, размер типа оп- ределяется числом элементов в списке: 1 байт (от 1 до 8 элементов), 2 байта (от 9 до 16

элементов), 3 байта (от 17 до 24 элементов), 4 байта (от 25 до 32 элементов) и 8 байт (от 33

до 64 элементов).

Календарные типы данных (табл. 14.4):

* + *DATE* – для хранения даты (формат *YYYY-MM-DD* для дат вида 2009-10-15 и формат

*YY-MM-DD* для дат вида 09-10-15);

* + *TIME* – для хранения времени суток (формат *HH:MM:SS*, где *HH* – часы, *MM* – минуты,

*SS* – секунды, например, 10:48:56);

* + *DATETIME*– для представления и даты, и времени суток;
  + *TIMESTAMP* – если в соответствующем столбце строки не указать конкретное значение или *NULL*, там будет записано время, когда соответствующая строка была создана или в по- следний раз изменена (в формате *DATETIME*);
    - *YEAR* – позволяет хранить только год. Таблица 14.4. Календарные типы дан- ных

|  |  |  |
| --- | --- | --- |
| **Тип** | **Объем памяти** | **Диапазон** |
| *DATE* | 3 байта | от '1000-01-01' до '9999-12-31' |
| *TIME* | 3 байта | от '-828:59:59' до '828:59:59' |
| *DATATIME* | 8 байт | от '1000-01-01 00:00:00' до '9999-12-31 00:00:00' |
| *TIMESTAMP (M)* | 4 байта | от '1970-01-01 00:00:00' до '2038-12-31 59:59:59' |
| *YEAR(2)*  *YEAR(4)* | 1 байт | формат *YY,* диапазон *–* от 1970 до 2069  формат *YYYY*, диапазон – от 1901 до 2155 |

Дни, месяцы, часы, минуты и секунды можно записывать как с ведущим нулем, так и без него. Например, все следующие записи идентичны:

'2009-04-06 02:04:08' '2009-4-06 02:04:08' '2009-4-6 02:04:08'

'2009-4-6 2:04:08' '2009-4-6 2:4:08' '2009-4-6 2:4:8'

В качестве разделителя между годами, месяцами, днями, часами, минутами, секун- дами может выступать любой символ, отличный от цифры. Так, следующие значения иден- тичны:

'09-12-31 11:30:45' '09.12.31 11+30+45' '09/12/31 11\*30\*45'

При указании времени после секунд через точку можно указать микросекунды, т. е. использовать расширенный формат вида *HH:MM:SS.FFFFFF*, например '10:25:14.000001'. Кроме того, можно использовать краткие форматы *НН:ММ* и *НН* (вместо пропущенных ве- личин будут подставлены нулевые значения).

Если время задается в недопустимом формате, то в поле записывается нулевое зна- чение. Нулевое значение присваивается полям временного типа по умолчанию, когда им не присваивается инициирующее значение (табл. 14.5).

Таблица 14.5. Значения времен полей временного типа

|  |  |
| --- | --- |
| **Тип** | **Нулевое значение** |
| *DATE* | '0000-00-00' |
| *TIME* | '00:00:00' |
| *DATATIME* | '0000-00-00 00:00:00' |
| *TIMESTAMP* | 00000000000000 |
| *YEAR* | 0000 |

Формат *TIMESTAMP* совпадает с *DATETIME*, но во внутреннем представлении дата хранится как число секунд, прошедших с полуночи 1 января 1970 г. (такое исчисление при- нято в операционной системе UNIX, а дата 01.01.1970 считается началом эпохи UNIX и днем рождения операционной системы).

Если в таблице несколько столбцов *TIMESTAMP*, при модификации записи текущее время будет записываться только в один из них (первый). Можно явно указать столбец, ко- торому необходимо назначать текущую дату при создании или изменении записи. Чтобы поля принимали текущую дату при создании записи, следует после определения столбца добавить *DEFAULT CURRENT\_TIMESTAMP*. Если текущее время должно выставляться при модификации записи, при использовании оператора *UPDATE* следует добавить *ON UPDATE CURRENT\_TIMESTAMP*.

Тип данных *NULL* используется, когда информации недостаточно и для части дан- ных нельзя определить, какое значение они примут. Для указания того, что поле может при- нимать неопределенное значение, в определении столбца после типа данных следует ука- зать ключевое слово *NULL.* Если поле не должно принимать значение *NULL*, следует ука- зать ключевое слово *NOT NULL*.

Рекомендации по выбору типа данных.

* + Обработка числовых данных происходит быстрее строковых. Так как типы *ENUM* и *SET* имеют внутреннее числовое представление, им следует отдавать предпочтение перед дру- гими видами строковых данных, если это возможно.
  + Производительность можно увеличить за счет представления строк в виде чисел. Пример – преобразование IP-адреса из строки в *BIGINT*. Это позволит уменьшить размер таблицы и значительно увеличить скорость при сортировке и выборке данных, но потребует дополни- тельных преобразований.
  + Базы данных хранятся на жестком диске, и чем меньше места они занимают, тем быстрее происходит поиск и извлечение. Если есть возможность, следует выбирать типы данных, занимающие меньше места.
  + Типы фиксированной длины обрабатываются быстрее типов переменной длины, т. к. в по- следнем случае при частых удалениях и модификациях таблицы происходит ее фрагмента- ция.
  + Если применение столбцов с данными переменной длины неизбежно, для дефрагментации таблицы следует применять команду *OPTIMIZE TABLE.*
  + **Обеспечение ссылочной целостности**. Задается конструкцией:

*FOREIGN KEY [name\_key] (col1, ... ) REFERENCES tbl (tbl\_col, ... )*

*[ON DELETE {CASCADE | SET NULL | NO ACTION | RESTRICT | SET DEFAULT}] [ON UPDATE {CASCADE | SET NULL | NO ACTION | RESTRICT | SET DEFAULT}]*

Конструкция позволяет задать внешний ключ с необязательным именем *name\_key* на столбцах, которые задаются в круглых скобках (один или несколько). Ключевое слово *REFERENCES* указывает таблицу *tbl*, на которую ссылается внешний ключ, в круглых скоб- ках указываются имена столбцов. Необязательные конструкции *ON DELETE* и *ON UPDATE* позволяют задать поведение СУБД при удалении и обновлении строк из таблицы-предка. Параметры, следующие за этими ключевыми словами, имеют следующие значения:

* + *CASCADE –* при удалении или обновлении записи в таблице-предке, содержащей первич- ный ключ, записи со ссылками на это значение в таблице-потомке удаляются или обновля- ются автоматически;
  + *SET NULL –* при удалении или обновлении записи в таблице-предке, содержащей первич- ный ключ, в таблице-потомке значения внешнего ключа, ссылающегося на таблицу-предка, устанавливаются в *NULL*;
  + *NO ACTION –* при удалении или обновлении записей, содержащих первичный ключ, с таб- лицей-потомком никаких действий не производится;
  + *RESTRICT –* если в таблице-потомке имеются записи, ссылающиеся на первичный ключ таблицы-предка, при удалении или обновлении записей с таким первичным ключом воз- вращается ошибка;
  + *SET DEFAULT* – согласно стандарту *SQL*, при удалении или обновлении первичного ключа в таблице-потомке для ссылающихся на него записей в поле внешнего ключа должно уста- навливаться значение по умолчанию (в MySQL это ключевое слово зарезервировано, но не обрабатывается).

**Создание индексов**. Индексы играют большую роль в БД, т. к. это основной способ ускорения их работы. Записи в таблице располагаются хаотически. Чтобы найти нужную запись, необходимо сканировать всю таблицу, на что уходит много времени. Идея индексов состоит в том, чтобы создать для столбца копию, которая постоянно будет поддерживаться в отсортированном состоянии. Это позволяет быстро осуществлять поиск по такому столбцу.

Все необходимые индексы формируются при создании таблицы. Индексированы бу- дут все столбцы, объявленные как *PRIMARY KEY, KEY, UNIQUE* или *INDEX*. Индекс также можно добавить с помощью оператора *CREATE INDEX*. Перед выполнением оператор пре- образуется в оператор *ALTER TABLE*. Например, создание индекса с именем *name* на основе поля *u\_name* из таблицы *users*:

*CREATE INDEX name ON users (u\_name);*

Перед ключевым словом *INDEX* может присутствовать *UNIQUE*, требующее уни- кальности ограничения.

Корректность таблиц в БД можно проверить с помощью оператора

*SHOW TABLES;*

Более подробную информацию о структуре таблицы дает команда

*DESCRIBE имя\_таблицы;*

**Переименование БД**. Специального оператора переименования БД нет, но можно переименовать каталог БД в системном каталоге (*…\DATA*).

**Удаление БД**. Удалить всю БД вместе с ее содержимым можно командой:

*DROP DATABASE [IF EXISTS] имя\_базы\_данных;*

**Удаление таблиц и индексов**. Удалить таблицу можно с помощью оператора:

*DROP TABLE [IF EXISTS] имя\_таблицы;*

Удалить индекс можно с помощью оператора:

*DROP INDEX имя\_индекса ON имя\_таблицы;*

**Изменение структуры таблиц.** Изменить структуру существующей таблицы можно с помощью оператора *ALTER TABLE*. Например, можно создать индекс *name* для таблицы *users* следующим образом: *ALTER TABLE users ADD INDEX name (u\_name);*

Оператор *ALTER TABLE* является исключительно гибким, поэтому он имеет огром- ное множество дополнительных ключевых слов.

## Задание для практической работы

**При выполнении практической работы необходимо для заданной предметной области средствами MySQL:**

* + создать базу данных;
  + создать таблицы, определить поля таблиц, индексы;
  + определить связи между таблицами и ограничения целостности;
  + составить отчет по практической работе.

Создайте базу данных *book* Интернет-магазина, торгующего компьютерной литера- турой. В базе данных должна поддерживаться следующая информация:

* + тематические каталоги, по которым сгруппированы книги;
  + предлагаемые книги (название, автор, год издания, цена, имеющееся на складе количество);
  + зарегистрированные покупатели (имя, отчество, фамилия, телефон, адрес электронной по- чты, статус – авторизованный, неавторизованный, заблокированный, активный с хорошей кредитной историей);
  + покупки, совершенные в магазине (время совершения покупки, число приобретенных эк- земпляров книги).

Логическая модель данных предметной области в стандарте IDEF1X представлена на рис.

14.1. Выделены сущности *КАТАЛОГ, КНИГА, КЛИЕНТ, ЗАКАЗ*, между которыми установ- лены не идентифицирующие связи мощностью oдин-ко-многим, определенные специфи- кой предметной области.



Рисунок 14.1. Логическая модель данных предметной области

Физическая модель данных предметной области в стандарте IDEF1X для целевой СУБД MySQL представлена на рис. 14.2.

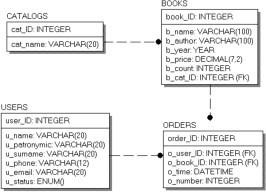


Рисунок 14.2. Физическая модель предметной области База данных *book* состоит из четырех таблиц:

* + *catalogs* – список торговых каталогов;
  + *books* – список предлагаемых книг;
  + *users* – список зарегистрированных пользователей магазина;
    - *orders* – список заказов (осуществленных сделок). Таблица *catalogs* состоит из двух полей:
  + *cat\_ID* – уникальный код каталога;
  + *cat\_name* – имя каталога.

Оба поля должны быть снабжены атрибутом *NOT NULL*, поскольку неопреде- ленное значение для них недопустимо.

Таблица *books* состоит из семи полей:

* + *book\_ID* – уникальный код книги;
  + *b\_name* – название книги;
  + *b\_author* – автор книги;
  + *b\_year* – год издания;
  + *b\_price* – цена книги;
  + *b\_count* – количество книг на складе;
  + *b\_cat\_ID* – код каталога из таблицы *catalogs*.

Цена книги *b\_price* и количество экземпляров на складе *b\_count* могут иметь атрибут *NULL*. На момент доставки часто неизвестны количество товара и его цена, но отразить факт наличия товара в прайс-листе необходимо.

Поле *b\_cat\_ID* устанавливает связь между таблицами *catalogs* и *books*. Это поле должно быть объявлено как внешний ключ (FK) с правилом каскадного удаления и обнов- ления. Обновление таблицы *catalogs* вызовет автоматическое обновление таблицы *books*. Удаление каталога в таблице *catalogs* приведет к автоматическому удалению всех записей в таблице *books*, соответствующих каталогу.

Таблица *users* состоит из семи полей:

* + *user\_ID* – уникальный код покупателя;
  + *u\_name* – имя покупателя;
  + *u\_patronymic* – отчество покупателя;
  + *u\_surname* – фамилия покупателя;
  + *u\_phone* – телефон покупателя (если имеется);
  + *u\_email* – e-mail покупателя (если имеется);
  + *u\_status* – статус покупателя.

Статус покупателя представлен полем типа *ENUM*, которое может принимать одно из четырех значений:

* + *active* – авторизованный покупатель, который может осуществлять покупки через Интернет;
  + *passive* – неавторизованный покупатель (значение по умолчанию), который осуществил процедуру регистрации, но не подтвердил ее и пока не может осуществлять покупки через Интернет, однако ему доступны каталоги для просмотра;
  + *lock –* заблокированный покупатель, не может осуществлять покупки и просматривать ка- талоги магазина;
  + *gold* – активный покупатель с хорошей кредитной историей, которому предоставляется скидка при следующих покупках в магазине.

Поля *u\_phone* и *u\_email* могут быть снабжены атрибутом *NULL*. Остальные поля должны получить атрибут *NOT NULL*.

Таблица *orders* включает пять полей:

* + *order\_ID* – уникальный номер сделки;
  + *o\_user\_ID* – номер пользователя из таблицы *users*;
  + *o\_book\_ID* – номер товарной позиции из таблицы *books*;
  + *o\_time* – время совершения сделки;
  + *o\_number* – число приобретенных товаров.

Поля таблицы *orders* должны быть снабжены атрибутом *NOT NULL*, т. к. при совер- шении покупки вся информация должна быть занесена в таблицу.

В таблице *orders* устанавливается связь с таблицами *users* (за счет поля *o\_user\_ID*) и *books* (за счет поля *o\_book\_ID*). Эти поля объявлены как внешние ключи (FK) с правилом каскадного удаления и обновления. Обновление таблиц *users* и *books* приведет к автомати- ческому обновлению таблицы *orders*. Удаление любого пользователя в таблице *users* при- ведет к автоматическому удалению всех записей в таблице *orders*, соответствующих этому пользователю.

Операторы создания БД *book* имеют следующий вид (целесообразно создать в *Блокноте*

текстовый файл и записать туда эти операторы).

*DROP DATABASE IF EXISTS book; CREATE DATABASE book;*

*USE book;*

*CREATE TABLE catalogs (*

*cat\_ID int(6) NOT NULL AUTO\_INCREMENT,*

*cat\_name varchar(20) NOT NULL, PRIMARY KEY (cat\_ID)*

*) TYPE=InnoDB; CREATE TABLE books (*

*book\_ID int(6) NOT NULL AUTO\_INCREMENT,*

*b\_name varchar(100) NOT NULL, b\_author varchar(100) NOT NULL, b\_year year NOT NULL, b\_price decimal(7,2) NULL default '0.00', b\_count int(6) NULL default '0', b\_cat\_ID int(6) NOT NULL default '0', PRIMARY KEY (book\_ID),*

*FOREIGN KEY (b\_cat\_ID) REFERENCES catalogs(cat\_ID) ON DELETE CASCADE ON UPDATE CASCADE*

*) TYPE=InnoDB; CREATE TABLE users (*

*user\_ID int(6) NOT NULL AUTO\_INCREMENT,*

*u\_name varchar(20) NOT NULL, u\_patronymic varchar(20) NOT NULL, u\_surname varchar(20) NOT NULL, u\_phone varchar(12) NULL,*

*u\_email varchar(20) NULL,*

*u\_status ENUM ('active','passive','lock','gold') default 'passive', PRIMARY KEY (user\_ID)*

*) TYPE=InnoDB;*

*CREATE TABLE orders (*

*order\_ID int(6) NOT NULL AUTO\_INCREMENT,*

*o\_user\_ID int NOT NULL, o\_book\_ID int NOT NULL,*

*o\_time datetime NOT NULL default '0000-00-00 00:00:00', o\_number int(6) NOT NULL default '0',*

*PRIMARY KEY (order\_ID),*

*FOREIGN KEY (o\_book\_ID) REFERENCES books(book\_ID) ON DELETE CASCADE ON UPDATE CASCADE,*

*FOREIGN KEY (o\_user\_ID) REFERENCES users(user\_ID) ON DELETE CAS- CADE ON UPDATE CASCADE*

*)TYPE=InnoDB**;*

## Практическая работа №14 Задание значений и ограничений поля. Проверка введен- ного в поле значения. Отображение данных числового типа и типа дата

**Цель работы:** овладение практическими навыками обработки табличных данных

## Вставка, удаление и обновление данных

После создания БД и таблиц перед ра**з**работчиком встает задача заполнения

таблиц

данными. В реляционных БД традиционно применяют три подхода:

* однострочный оператор *INSERT* – добавляет в таблицу новую запись;
* многострочный оператор *INSERT* – добавляет в таблицу несколько записей;
* пакетная загрузка *LOAD DATA INFILE* – добавление данных из файла.

**Вставка данных с помощью оператора *INSERT***. Однострочный оператор *INSERT*

может использоваться в нескольких формах. Упрощенный синтаксис первой формы:

*INSERT [IGNORE] [INTO] имя\_таблицы [(имя\_столбца, ... )] VALUES (выражение, ... );*

Оператор вставляет новую запись в таблицу *имя\_таблицы*. Значения полей записи перечисляются в списке *(выражение, ... )*. Порядок следования столбцов задается списком *(имя\_столбца, ... )*. Список столбцов *(имя\_столбца, ... )* позволяет менять порядок следова- ния столбцов при добавлении.

Первичный ключ таблицы является уникальным, и попытка добавить уже существу- ющее значение приведет к ошибке. Чтобы новые записи с дублирующим ключом отбрасы- вались без генерации ошибки, следует добавить после оператора *INSERT* ключевое слово *IGNORE*.

Другая форма оператора *INSERT* предполагает использование слова *SET*: *INSERT [IGNORE] [INTO] имя\_таблицы*

*SET имя\_столбца1 = выражение1, имя\_столбца2 = выражение2, ... ;*

Оператор заносит в таблицу *имя\_таблицы* новую запись, столбец *имя\_столбца* в ко- торой получает значение *выражение*.

Многострочный оператор *INSERT* совпадает по форме с однострочным операто- ром, но после ключевого слова *VALUES* добавляется через запятую несколько списков *(вы- ражение, ... )*.

Практические примеры использования оператора *INSERT* для заполнения учебной БД *book*

см. ниже, в пункте «Пример выполнения работы».

**Удаление данных.** Для удаления записей из таблиц предусмотрены:

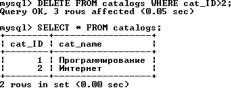
* оператор *DELETE;*
* оператор *TRUNCATE TABLE.*

Оператор *DELETE* имеет следующий синтаксис:

*DELETE FROM имя\_таблицы*

*[WHERE условие] [ORDER BY имя\_поля] [LIMIT число\_строк];*

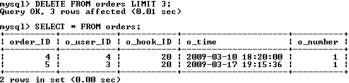
Оператор удаляет из таблицы *имя\_таблицы* записи, удовлетворяющие условию. В следующем примере из таблицы *catalogs* удаляются записи, имеющие значение первичного ключа *catalog\_id* больше двух.



Если в операторе отсутствует условие *WHERE*, удаляются все записи таблицы.



Ограничение *LIMIT* позволяет задать максимальное число записей, которые могут быть удалены. Следующий запрос удаляет все записи таблицы *orders*, но не более 3 записей.



Конструкция *ORDER BY* обычно применяется вместе с ключевым словом *LIMIT*. Например, если необходимо удалить 20 первых записей таблицы, то производится сорти- ровка по полю типа *DATETIME* – тогда в первую очередь будут удалены самые старые записи.

Оператор *TRUNCATE TABLE* полностью очищает таблицу и не допускает условного удаления. Он аналогичен оператору *DELETE* без условия *WHERE* и ограничения *LIMIT*. Удаление происходит гораздо быстрее, т. к. осуществляется не перебор записей, а полное очищение таблицы.



**Обновление данных.** Обновление данных (изменение значений полей в существу- ющих записях) обеспечивают:

* + оператор *UPDATE*;
  + оператор *REPLACE*.

Оператор *UPDATE* позволяет обновлять отдельные поля в существующих записях.

Имеет следующий синтаксис

*UPDATE [IGNORE] имя\_таблицы*

*SET имя\_столбца1= выражение1 [, имя\_столбца2 = выражение2 … ] [WHERE условие] [ORDER BY имя\_поля ] [LIMIT число\_строк] ;*

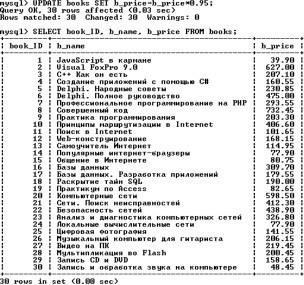
После ключевого слова *UPDATE* указывается таблица, которая изменяется. В пред- ложении *SET* указывается, какие столбцы обновляются и устанавливаются их новые значе- ния. Необязательное условие *WHERE* позволяет задать критерий отбора строк (обновляться будут только строки, удовлетворяющие условию).

Если указывается необязательное ключевое слово *IGNORE*, то команда обновления не будет прервана, даже если при обновлении возникнет ошибка дублирования ключей. Строки, породившие конфликтные ситуации, обновлены не будут.

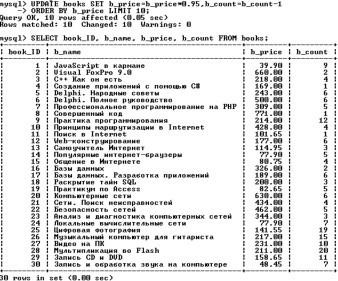
Запрос, изменяющий в таблице *catalogs* «Сети» на «Компьютерные сети».



Обновлять можно всю таблицу. Пусть требуется уменьшить на 5 % цену на все книги. Для этого следует старую цену в рублях умножить на 0,95.



Инструкции *LIMIT* и *ORDER BY* позволяют ограничить число изменяемых записей. При этом за один запрос можно обновить несколько столбцов таблицы. Например, необхо- димо в таблице *books* для десяти самых дешевых товарных позиций уменьшить количество книг на складе на единицу, а цену – на 5 %.



Оператор *REPLACE* работает как оператор *INSERT,* за исключением того, что старая запись с тем же значением индекса *UNIQUE* или *PRIMARY KEY* перед внесением новой будет удалена. Если не используются индексы *UNIQUE* или *PRIMARY KEY,* то применение оператора *REPLACE* не имеет смысла.

Синтаксис оператора *REPLACE* аналогичен синтаксису оператора *INSERT*: *REPLACE [INTO] имя\_таблицы [(имя\_столбца, ... )]*

*VALUES (выражение, ... )*В таблицу вставляются значения, определяемые в списке после ключевого слова *VALUES*. Задать порядок столбцов можно при помощи необязательного списка, следующего за именем таблицы. Как и оператор *INSERT*, оператор *REPLACE* до- пускает многострочный формат.

### Задание для практической работы

При выполнении практической работы необходимо для заданной предметной обла- сти средствами MySQL:

* + заполнить согласованными данными таблицы БД;
  + при необходимости исправить введенную информацию;
    - составить отчет по практической работе. Операторы заполнения БД *book*

имеют следующий вид. *USE book; SET CHARACTER SET cp1251;*

*DELETE FROM catalogs;*

*INSERT INTO catalogs VALUES (1,'Программирование'); INSERT INTO catalogs VALUES (2,'Интернет');*

*INSERT INTO catalogs VALUES (3,'Базы данных'); INSERT INTO catalogs VALUES (4,'Сети'); INSERT INTO catalogs VALUES (5,'Мультимедиа'); DELETE FROM books;*

*INSERT INTO books VALUES (1,'JavaScript в кармане','Рева О.Н.', 2008, 42.00, 10, 1);*

*INSERT INTO boos VALUES (2,'Visual FoxPro 9.0','Клепинин В.Б.', 2007, 660.00, 2, 1); INSERT*

*INTO books VALUES (3,'C++ Как он есть','Тимофеев В.В.',2009, 218.00, 4, 1);*

*INSERT INTO books VALUES (4,'Создание приложений с помощью C#','Фаронов В.В.', 2008, 169.00, 1, 1);*

*INSERT INTO books VALUES (5,'Delphi. Народные советы','Шкрыль А.А.',2007,243.00,6,1); INSERT INTO books VALUES (6,'Delphi. Полное руководство','Сухарев М.',2008,500.00,6,1); INSERT INTO books VALUES (7,'Профессиональное программирование на PHP', 'Шлосснейгл Дж.', 2006, 309.00, 5, 1);*

*INSERT INTO books VALUES (8,'Совершенный код','Макконнелл С.', 2007, 771.00, 1, 1); INSERT INTO books VALUES (9,'Практика программирования','Керниган Б.', 2004, 214.00,12, 1);*

*INSERT INTO books VALUES (10,'Принципы маршрутизации в Internet','Хелеби С.', 2001,428.00, 4, 2);*

*INSERT INTO books VALUES (11,'Поиск в Internet','Гусев В.С.',2004,107.00,2,2);*

*INSERT INTO books VALUES (12,'Web-конструирование','Дуванов А.А.', 2003, 177.00, 6, 2); INSERT INTO books VALUES (13,'Самоучитель Интернет','Константинов Ю.П.', 2009, 121.00, 4, 2);*

*INSERT INTO books VALUES (14,'Популярные интернет-браузеры','Маринин С.А.', 2007, 82.00, 6, 2);*

*INSERT INTO books VALUES (15,'Общение в Интернете','Экслер А.', 2006, 85.00, 5, 2);*

*INSERT INTO books VALUES (16,'Базы данных','Малыхина М.П.', 2006, 326.00, 2, 3); INSERT*

*INTO books VALUES (17,'Базы данных. Разработка приложений','Рудикова Л.В.', 2006, 189.00, 6, 3);*

*INSERT INTO books VALUES (18,'Раскрытие тайн SQL','Оппель Э.', 2007, 200.00, 3, 3);*

*INSERT INTO books VALUES (19,'Практикум по Access','Золотова С.И.', 2007, 87.00, 6, 3);*

*INSERT INTO books VALUES (20,'Компьютерные сети','Танненбаум Э.', 2007, 630.00, 6, 4); INSERT INTO books VALUES (21,'Сети. Поиск неисправностей','Бигелоу С.', 2005, 434.00, 4, 4);*

*INSERT INTO books VALUES (22,'Безопасность сетей','Брегг Р.', 2006, 462.00, 5, 4); INSERT INTO books VALUES (23,'Анализ и диагностика компьютерных сетей', 'Хогдал Дж.', 2001, 344.00, 3, 4);*

*INSERT INTO books VALUES (24,'Локальные вычислительные сети', 'Епанешников А.' , 2005, 82.00, 8, 4);*

*INSERT INTO books VALUES (25,'Цифровая фотография','Надеждин Н.', 2004, 149.00,20,5);*

*INSERT INTO books VALUES (26,'Музыкальный компьютер для гитариста', 'Петелин*

*.Ю.', 2004, 217.00, 15, 5);*

*20, 5);*

*INSERT INTO books VALUES (27,'Видео на ПК','Федорова А.',2003,231.00,10,5);*

*INSERT INTO books VALUES (28,'Мультипликация во Flash','Киркпатрик Г.', 2006, 211.00,*

*INSERT INTO books VALUES (29,'Запись CD и DVD','Гультяев А.К.', 2003, 167.00, 12, 5);*

*INSERT INTO books VALUES (30,'Запись и обработка звука на компьютере', 'Лоянич А.А.',2008, 51.00, 8, 5);*

*DELETE FROM users;*

*INSERT INTO users VALUES*

*(1,'Александр','Валерьевич','Иванов','58-98-78',* [*'ivanov@email.ru',*](mailto:%27ivanov@email.ru) *'ac-*

*tive');*

*INSERT INTO users VALUES (2,'Сергей','Иванович','Лосев','90-57-77', 'lo-*

[*sev@email.ru',*](mailto:sev@email.ru) *'passive');*

*INSERT INTO users VALUES*

*(3,'Игорь','Николаевич','Симонов','95-66-61',* [*'simonov@email.ru',*](mailto:%27simonov@email.ru) *'ac-*

*tive');*

*INSERT INTO users VALUES (4,'Максим','Петрович','Кузнецов',NULL, 'kuz-*

[*netsov@email.ru',*](mailto:netsov@email.ru) *'active');*

*INSERT INTO users VALUES (5,'Анатолий','Юрьевич','Петров', NULL, NULL, 'lock'); IN-*

*SERT INTO users VALUES*

*(6,'Александр','Александрович','Корнеев','89-78-36',* [*'korneev@email.ru',*](mailto:%27korneev@email.ru) *'gold'); DELETE FROM orders;*

*INSERT INTO orders VALUES (1,3,8,'2009-01-04 10:39:38',1);*

*INSERT INTO orders VALUES (2,6,10,'2009-02-10 09:40:29',2);*

*INSERT INTO orders VALUES (3,1,20,'2009-02-18 13:41:05',4);*

*INSERT INTO orders VALUES (4,4,20,'2009-03-10 18:20:00',1);*

*INSERT INTO orders VALUES (5,3,20,'2009-03-17 19:15:36',1)**;*

## Практическая работа №15 Создание и модификация таблиц БД. Выборка дан- ных из БД. Модификация содержимого БД

**Цель работы:** овладение практическими навыками по модификации таблиц БД, со- здание запросов к БД

### Создание простых запросов на выборку

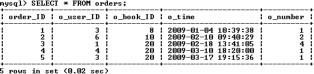
Для выполнения запросов (извлечения строк из одной или нескольких таблиц БД) используется оператор *SELECT*. Результатом запроса всегда является таблица. Результаты запроса могут быть использованы для создания новой таблицы. Таблица, полученная в ре- зультате запроса, может стать предметом дальнейших запросов.

Общая форма оператора *SELECT*:

*SELECT столбцы FROM таблицы [WHERE условия]*

*[GROUP BY группа [HAVING групповые\_условия] ] [ORDER BY имя\_поля] [LIMIT пределы];*

Оператор *SELECT* имеет много опций. Их можно использовать или не использовать, но они должны указываться в том порядке, в каком они приведены. Если требуется вывести все столбцы таблицы, необязательно перечислять их после ключевого слова *SELECT*, до- статочно заменить этот список символом \*.



Список столбцов в операторе *SELECT* используют, если нужно изменить порядок следования столбцов в результирующей таблице или выбрать часть столбцов.



**Условия выборки**. Гораздо чаще встречается ситуация, когда необходимо изменить количество выводимых строк. Для выбора записей, удовлетворяющих определенным кри- териям поиска, можно использовать конструкцию *WHERE*.

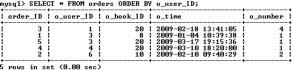


В запросе можно использовать ключевое слово *DISTINCT*, чтобы результат не содер-



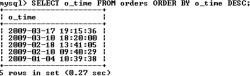
жал повторений уже имеющихся значений, например:

**Сортировка**. Результат выборки – записи, расположенные в том порядке, в котором они хранятся в БД. Чтобы отсортировать значения по одному из столбцов, необходимо по- сле конструкции *ORDER BY* указать этот столбец, например:



Сортировку записей можно производить по нескольким столбцам (их следует ука- зать после слов *ORDER BY* через запятую). Число столбцов, указываемых в конструкции *ORDER BY*, не ограничено.

По умолчанию сортировка производится в прямом порядке (записи располагаются от наименьшего значения поля сортировки до наибольшего). Обратный порядок сорти-



ровки реализуется с помощью ключевого слова *DESC*:

Для прямой сортировки существует ключевое слово *ASC*, но так как записи сортиру- ются в прямом порядке по умолчанию, данное ключевое слово опускают.

**Ограничение выборки**. Результат выборки может содержать тысячи записей, вы- вод и обработка которых занимают значительное время. Поэтому информацию часто раз- бивают на страницы и предоставляют ее пользователю частями. Постраничная навигация используется при помощи ключевого слова *LIMIT*, за которым следует число выводимых записей. Следующий запрос извлекает первые 5 записей, при этом осуществляется обратная



сортировка по полю *b\_count*:

Для извлечения следующих пяти записей используется ключевое слово *LIMIT* с двумя цифрами. Первая указывает позицию, начиная с которой необходимо вернуть резуль- тат, вторая цифра – число извлекаемых записей, например:



При определении смещения нумерация строк начинается с нуля (поэтому в послед- нем примере для шестой строки указано смещение 5).

**Группировка записей**. Конструкция *GROUP ВУ* позволяет группировать извлекае- мые строки. Она полезна в комбинации с функциями, применяемыми к группам строк. Эти функции (табл. 16.1) называются агрегатами (суммирующими функциями) и вычисляют одно значение для каждой группы, создаваемой конструкцией *GROUP BY*. Функции позво- ляют узнать число строк в группе, подсчитать среднее значение, получить сумму значений столбцов. Результирующее значение рассчитывается для значений, не равных *NULL* (ис- ключение – функция *COUNT(\*)*). Допустимо использование этих функций в запросах без группировки (вся выборка – одна группа).

Пример использования функции *COUNT( )*, которая возвращает число строк в таб- лице, значения указанного столбца для которых отличны от *NULL*:

Таблица 16.1. Агрегатные функции

|  |  |
| --- | --- |
| **Обозначение** | **Описание** |
| *AVG ([DISTINCT]*  *expr)* | Возвращает среднее значение аргумента *expr*. В качестве аргумента обычно выступает имя столбца. Необяза- тельное слово*DISTINCT* позволяет  обрабатывать только уникальные значения столбца *expr* |
| *COUNT ( )* | Подсчитывает число записей и имеет несколько форм. Форма *COUNT (выражение)* возвращает число записей в таблице, поле *выражение* для которых не равно *NULL*. Форма *COUNT(\*)* возвращает общее число строк в таблице независимо от того, принимает какое-либо поле значение *NULL* или нет. Форма *COUNT (DISTINCT выражение1, выражение2, ... )* позво- ляет использовать ключевое слово *DISTINCT*, которое позволяет под- считать  только уникальные значения столбца |
| *MIN ([DISTINCT]*  *expr)* | Возвращает минимальное значение среди всех непустых значений  выбранных строк в столбце *expr*. Необязательное слово *DISTINCT* позво- ляет обрабатывать только уникальные значения столбца *expr* |
| *MAX ([DISTINCT]*  *expr)* | Возвращает максимальное значение среди всех непустых значений выбранных строк в столбце *expr*. Необязательное слово *DISTINCT* позво- ляет  обрабатывать только уникальные значения столбца *expr* |
| *STD (expr)* | Возвращает стандартное среднеквадратичное отклонение в аргументе  *expr* |
| *STDDEV\_SAMP*  *(expr)* | Возвращает выборочное среднеквадратичное отклонение в аргументе  *expr* |
| *SUM ([DISTINCT]*  *expr)* | Возвращает сумму величин в столбце *expr*. Необязательное слово *DIS- TINCT*  позволяет обрабатывать только уникальные значения столбца *expr* |

Использование ключевого слова *DISTINCT* с функцией *COUNT( )* позволяет вернуть число уникальных значений *b\_cat\_ID* в таблице *books*, например:

В *SELECT*-запросе столбцу можно назначить новое имя с помощью оператора *AS*. Например, результату функции *COUNT( )* присваивается псевдоним *total*:



Использование функций в конструкции *WHERE* приведет к ошибке. В следующем примере показана попытка извлечения из таблицы *catalogs* записи с максимальным значе- нием поля *cat\_ID*:



Решение задачи следует искать в использовании конструкции *ORDER BY*:

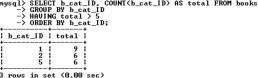
Для извлечения уникальных записей используют конструкцию *GROUP BY* с име- нем столбца, по которому группируется результат:



При использовании *GROUP BY* возможно использование условия *WHERE*:



Часто при задании условий требуется ограничить выборку по результату функции (например, выбрать каталоги, где число товарных позиций больше 5). Использование для этих целей конструкции *WHERE* приводит к ошибке. Для решения этой проблемы вместо ключевого слова *WHERE* используется ключевое слово *HAVING*, располагающееся за кон- струкцией *GROUP BY*:



Запрос, извлекающий уникальные значения столбца *b\_cat\_ID*, большие двух:



При этом в случае использования ключевого слова *WHERE* сначала производится выборка из таблицы с применением условия и лишь затем группировка результата, а в слу- чае использования ключевого слова *HAVING* сначала происходит группировка таблицы и лишь затем выборка с применением условия. Допускается использование условия *HAVING* без группировки *GROUP BY*.

**Использование функций**. Для решения специфических задач при выборке удобны встроенные функции MySQL. Большинство функций предназначено для использования в выражениях *SELECT* и *WHERE*. Существуют также специальные функции группировки для использования в выражении *GROUP BY* (см. выше).

Каждая функция имеет уникальное имя и может иметь несколько аргументов (пере- числяются через запятую в круглых скобках). Если аргументы отсутствуют, круглые скобки все равно следует указывать. Пробелы между именем функции и круглыми скобками недо- пустимы.

Число доступных для использования функций велико, в приложениях приведены наиболее полезные из них.

Пример использования функции, возвращающей версию сервера MySQL:

Отметим также возможность использования оператора *SELECT* без таблиц вообще.

В такой форме *SELECT* можно использовать как калькулятор:



Можно вычислить любое выражение без указания таблиц, получив доступ ко всему разнообразию математических и других операторов и функций. Возможность выполнять математические расчеты на уровне *SELECT* позволяет проводить финансовый анализ зна- чений таблиц и отображать полученные результаты в отчетах. Во всех выражениях MySQL (как в любом языке программирования) можно использовать скобки, чтобы контролировать порядок вычислений.

**Операторы**. Под операторами подразумеваются конструкции языка, которые про- изводят преобразование данных. Данные, над которыми совершается операция, называются операндами.

В MySQL используются три типа операторов:

* арифметические операторы;
* операторы сравнения;
* логические операторы.

*Арифметические операции.* В MySQL используются обычные арифметические опе- рации: сложение (+), вычитание (–), умножение (\*), деление (/) и целочисленное деление *DIV* (деление и отсечение дробной части). Деление на 0 дает безопасный результат *NULL*.

*Операторы сравнения.* При работе с операторами сравнения необходимо помнить о том, что, за исключением нескольких особо оговариваемых случаев, сравнение чего-либо со значением *NULL* дает в результате *NULL*. Это касается и сравнения значения *NULL* со значением *NULL*:



Корректнее использовать следующий запрос:

Поэтому следует быть предельно внимательными при работе с операторами сравне- ния, если операнды могут принимать значения *NULL*.

Наиболее часто используемые операторы сравнения приведены в табл. 16.2.

*Логические операторы.* MySQL поддерживает все обычные логические операции, которые можно использовать в выражениях. Логические выражения в MySQL могут при- нимать значения 1 (истина), 0 (ложь) или *NULL*.

Кроме того, следует учитывать, что MySQL интерпретирует любое ненулевое значе- ние, отличное от *NULL*, как значение «истина». Основные логические операторы приве- дены в табл. 8.

Таблица 16.2. Наиболее часто используемые операторы

|  |  |
| --- | --- |
| **Оператор** | **Значение** |
| **=** | Оператор равенства. Возвращает 1 (истина), если операнды равны, и 0 (ложь),  если не равны |

|  |  |
| --- | --- |
| <=> | Оператор эквивалентности. Аналогичен обычному равенству, но возвра- щает  только два значения: 1 (истина) и 0 (ложь). *NULL* не возвращает |
| <> | Оператор неравенства. Возвращает 1 (истина), если операнды не равны, и 0  (ложь), если равны |
| < | Оператор «меньше». Возвращает 1 (истина), если левый операнд меньше  правого, и 0 (ложь) – в противном случае |
| <= | Оператор «меньше или равно». Возвращает 1 (истина), если левый опе- ранд  меньше правого или они равны, и 0 (ложь) – в противном случае |
| > | Оператор «больше». Возвращает 1 (истина), если левый операнд больше  правого, и 0 (ложь) – в противном случае |
| >= | Оператор «больше или равно». Возвращает 1 (истина), если левый опе- ранд  больше правого или они равны, и 0 (ложь) – в противном случае |
| *n BETWEEN*  *min*  *AND max* | Проверка диапазона. Возвращает 1 (истина), если проверяемое значение  *n*  находится между *min* и *max*, и 0 (ложь) – в противном случае |
| *IS NULL* и  *IS NOT NULL* | Позволяют проверить, является ли значение значением *NULL* или нет |
| *n IN*  *(множество)* | Принадлежность к множеству. Возвращает 1 (истина), если проверяемое значение *n* входит в список, и 0 (ложь) – в противном случае. В каче- стве  множества может использоваться список литеральных значений или выражений или подзапрос |

Таблица 16.3. Логические операторы

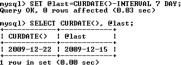
|  |  |  |
| --- | --- | --- |
| **Оператор** | **Пример** | **Значение** |
| *AND* | *n AND m* | Логическое *И*: истина *AND* истина = истина,  ложь *AND* любое = ложь. Все остальные выражения оценива- ются как *NULL* |
| *OR* | *n OR m* | Логическое *ИЛИ*: истина *OR* любое = истина,  *NULL OR* ложь = *NULL,*  *NULL OR NULL* = *NULL,* ложь *OR* ложь = ложь |
| *NOT* | *NOT n* | Логическое НЕТ: *NOT* истина = ложь, *NOT* ложь = истина.  *NOT NULL = NULL* |
| *XOR* | *n XOR m* | Логическое *исключающее ИЛИ*: истина *XOR* истина = ложь, ис- тина *XOR* ложь = истина, ложь *XOR* истина = истина, ложь *XOR* ложь = ложь,  *NULL XOR* любое *= NULL*, любое *XOR NULL = NULL* |

**Переменные SQL и временные таблицы**. Часто результаты запроса необходимо использовать в последующих запросах. Для этого полученные данные необходимо сохра- нить во временных структурах. Эту задачу решают переменные SQL и временные таблицы. Объявление переменной начинается с символа @, за которым следует имя переменной. Зна- чения переменным присваиваются посредством оператора *SELECT* С использованием опе- ратора присваивания := . Например:



Объявляется переменная @*total*, которой присваивается число записей в таблице *books*. Затем в рамках текущего сеанса в последующих запросах появляется возможность использования данной переменной. Переменная действует только в рамках одного сеанса соединения с сервером MySQL и прекращает свое существование после разрыва соедине- ния.

Переменные также могут объявляться при помощи оператора *SET*:



При использовании оператора *SET* в качестве оператора присваивания может высту- пать обычный знак равенства =. Оператор *SET* удобен тем, что он не возвращает результи- рующую таблицу. Не рекомендуется одновременно присваивать переменной некоторое значение и использовать эту переменную в одном запросе.

Переменная SQL позволяет сохранить одно промежуточное значение. Когда необхо- димо сохранить результирующую таблицу, прибегают к временным таблицам. Создание временных таблиц осуществляется при помощи оператора *CREATE TEMPORARY TABLE,* синтаксис которого ничем не отличается от синтаксиса оператора *CREATE TABLE*.

Временная таблица автоматически удаляется по завершении соединения с сервером, а ее имя действительно только в течение данного соединения. Это означает, что два разных клиента могут использовать временные таблицы с одинаковыми именами без конфликта друг с другом или с существующей таблицей с тем же именем.

### Задание для практической работы

1. Создайте простой запрос на выборку к таблице *books*, который выводит мак- симальную и минимальную цены товарных позиций, присваивая им соответственно псев- донимы *maximum* и *minimum*:

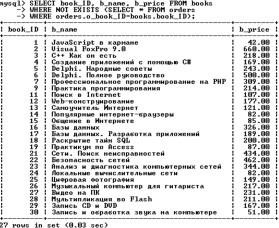


1. Создавайте простой запрос на выборку к таблице *books*, который выводит количество записей, соответствующих каждому из уникальных значений *b\_cat\_ID.* Для этого используем функцию *COUNT( )* вместе с выражением *GROUP BY*:



1. Создавайте многотабличный запрос на выборку, который выводит фамилии, имена и отчества покупателей магазина, сделавших менее двух покупок:



1. Создайте запрос на выборку с вложенным запросом, выводящим перечень книг, которые не заказывались покупателями:

## Практическая работа №16 Обработка транзакций. Использование функций за- щиты для БД

**Цель работы:** получение практических навыков обработки транзакций и защиты баз данных при помощи функций.

### Обработка транзакций

Изменения БД часто требуют выполнения нескольких запросов, например при по- купке в электронном магазине требуется добавить запись в таблицу заказов и уменьшить число товарных позиций на складе. В промышленных БД одно событие может затрагивать большее число таблиц и требовать многочисленных запросов.

Если на этапе выполнения одного из запросов происходит сбой, это может нарушить целостность БД (товар может быть продан, а число товарных позиций на складе не обнов- лено). Чтобы сохранить целостность БД, все изменения должны выполняться как единое целое. Либо все изменения успешно выполняются, либо, в случае сбоя, БД принимает со- стояние, которое было до начала изменений. Это обеспечивается средствами обработки транзакций.

*Транзакция* – последовательность операторов SQL, выполняющихся как единая опе- рация, которая не прерывается другими клиентами. Пока происходит работа с записями таблицы (обновление или удаление), никто другой не может получить доступ к этим запи- сям, т. к. MySQL автоматически блокирует доступ к ним.

Таблицы *ISAM, MyISAM* и *HEAP* не поддерживают транзакции. В настоящий момент их поддержка осуществляется только в таблицах *BDB* и *InnoDB*.

Транзакции позволяют объединять операторы в группу и гарантировать, что все опе- раторы группы будут выполнены успешно. Если часть транзакции выполняется со сбоем, результаты выполнения всех операторов транзакции до места сбоя отменяются, приводя БД к виду, в котором она была до выполнения транзакции.

По умолчанию MySQL работает в режиме автоматического завершения транзакций, т. е. как только выполняется оператор обновления данных, который модифицирует таблицу, изменения тут же сохраняются на диске. Чтобы объединить операторы в транзакцию, сле- дует отключить этот режим: *SET AUTOCOMMIT=0;*

После отключения режима для завершения транзакции необходимо ввести оператор

*COMMIT*, для отката – *ROLLBACK*.

Включить режим автоматического завершения транзакций для отдельной последо- вательности операторов можно оператором *START TRANSACTION*.

Для таблиц *InnoDB* есть операторы *SAVEPOINT* и *ROLLBACK TO SAVEPOINT*, кото- рые позволяют работать с именованными точками начала транзакции.





Оператор *SAVEPOINT* устанавливает именованную точку начала транзакции с име- нем *point1*. Оператор *ROLLBACK TO SAVEPOINT point1* откатывает транзакцию к состоя- нию, в котором находилась БД на момент установки именованной точки. Все точки сохра- нения транзакций удаляются, если выполняются операторы *COMMIT* или *ROLLBACK* без указания имени точки сохранения.

### Управление правами пользователей

СУБД MySQL является многопользовательской средой, поэтому для доступа к таб- лицам БД могут быть созданы различные учетные записи с разным уровнем привилегий. Учетной записи редактора можно предоставить привилегии на просмотр таблицы, добавле- ние новых записей и обновление уже существующих. Администратору БД можно предо- ставить более широкие полномочия (возможность создания таблиц, редактирования и уда- ления уже существующих). Для пользователя БД достаточно лишь просмотра таблиц.

Рассмотрим следующие вопросы:

* создание, редактирование и удаление учетных записей пользователей;
* назначение и отмена привилегий.

Учетная запись является составной и принимает форму *'username' @ 'host'*, где *username* – имя пользователя, a *host* – наименование хоста, с которого пользователь может обращаться к серверу. Например, записи *'root' @ '127.0.0.1'* и *'wet' @ '62.78.56.34'* означают, что пользователь с именем *root* может обращаться с хоста, на котором расположен сервер, a *wet* – только с хоста с IP- адресом 62.78.56.34.

IP-адрес 127.0.0.1 всегда относится к локальному хосту. Если сервер и клиент уста- новлены на одном хосте, то сервер слушает соединения по этому адресу, а клиент отправ- ляет на него SQL- запросы.

IP-адрес 127.0.0.1 имеет псевдоним *localhost*, поэтому учетные записи вида *'root' @ '127.0.0.1'* можно записывать в виде *'root' @ 'localhost'*.

Число адресов, с которых необходимо обеспечить доступ пользователю, может быть значительным. Для задания диапазона в имени хоста используется специальный символ "%". Так, учетная запись *'wet' @ '%'* позволяет пользователю *wet* обращаться к серверу MySQL с любых компьютеров сети.

Все учетные записи хранятся в таблице *user* системной базы данных с именем *mysql*.



После первой инсталляции содержимое таблицы *user* выглядит так, как показано в ли- стинге.

**Создание новой учетной записи**. Создать учетную запись позволяет оператор

*CREATE USER 'username' @ 'host' [IDENTIFIED BY [PASSWORD] 'пароль'];*

Оператор создает новую учетную запись с необязательным паролем. Если пароль не указан, в его качестве выступает пустая строка. Разумно хранить пароль в виде хэш-кода, полученного в результате необратимого шифрования. Чтобы воспользоваться этим меха- низмом авторизации, необходимо поместить между ключевым словом *IDENTIFIED BY* и паролем ключевое слово *PASSWORD*.

**Удаление учетной записи**. Удалить учетную запись позволяет оператор

*DROP USER 'username' @ 'host';*

**Изменение имени пользователя в учетной записи**. Осуществляется с помощью оператора

*RENAME USER старое\_имя TO новое\_имя;*

**Назначение привилегий**. Рассмотренные выше операторы позволяют создавать, удалять и редактировать учетные записи, но они не позволяют изменять привилегии пользователя – сооб- щать MySQL, какой пользователь имеет право только на чтение информации, какой на чтение и редактирование, а кому предоставлены права изменять структуру БД и созда- вать учетные записи.

Для решения этих задач предназначены операторы *GRANT* (назначает привилегии) и *REVOKE* (удаляет привилегии). Если учетной записи, которая показана в операторе *GRANT*, не существует, то она автоматически создается. Удаление всех привилегий с помо- щью оператора *REVOKE* не приводит к автоматическому уничтожению учетной записи. Для удаления пользователя необходимо воспользоваться оператором *DROP USER*.

В простейшем случае оператор *GRANT* выглядит следующим образом:



Данный запрос создает пользователя с именем *wet* и паролем *pass*, который может обращаться к серверу с локального хоста (*localhost*) и имеет все права (*ALL*) для всех баз данных (\*.\*). Если такой пользователь существует, то его привилегии будут изменены на *ALL*.

Вместо ключевого слова *ALL* можно использовать любое из ключевых слов, пред- ставленных в табл. 17.1. Ключевое слово *ON* в операторе *GRANT* задает уровень привиле- гий, которые могут быть заданы на одном из четырех уровней, представленных в табл. 10. Для таблиц можно установить только следующие типы привилегий: *SELEСТ, INSERT, UPDATE, DELETE, CREATE, DROP, GRANT OPTION, INDEX* и *ALTER*. Это следует учиты-

вать при использовании конструкции *GRANT ALL*, которая назначает привилегии на теку- щем уровне. Так, запрос уровня базы данных *GRANT ALL ON db.\** не предоставляет ника- ких глобальных привилегий.

**Отмена привилегий**. Для отмены привилегий используется оператор *REVOKE*:

Оператор *REVOKE* отменяет привилегии, но не удаляет учетные записи, для их уда- ления необходимо воспользоваться оператором *DROP USER*.

Таблица 17.1. Вместо ключевого слова ALL

|  |  |  |
| --- | --- | --- |
| **Привилегия** | | **Операция, разрешенная привилегией** |
| *ALL [PRIVILEGES]* | | Комбинация всех привилегий, за исключением привилегии  *GRANT OPTION*, которая задается отдельно |
| *ALTER* | | Позволяет редактировать таблицу с помощью оператора *AL- TER*  *TABLE* |
| *ALTER ROUTINE* | | Позволяет редактировать или удалять хранимую процедуру |
| *CREATE* | | Позволяет создавать таблицу при помощи оператора *CREATE TA-*  *BLE* |
| *CREATE ROUTINE* | | Позволяет создавать хранимую процедуру |
| *CREATE*  *TABLES* | *TEMPORARY* | Позволяет создавать временные таблицы |
| *CREATE USER* | | Позволяет работать с учетными записями c помощью *CREATE*  *USER, DROP USER, RENAME USER* и *REVOKE ALL PRIVILEGES* |
| *CREATE VIEW* | | Позволяет создавать представление с помощью оператора *CRE- ATE*  *VIEW* |
| *DELETE* | | Позволяет удалять записи при помощи оператора *DELETE* |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *DROP* | Позволяет удалять таблицы при помощи оператора *DROP TABLE* | | | |
| *EXECUTE* | Позволяет выполнять хранимые процедуры | | | |
| *INDEX* | Позволяет работать с индексами, операторы  *CREATE INDEX* и *DROP INDEX* | в | частности, | использовать |
| *INSERT* | Позволяет добавлять в таблицу новые записи оператором *INSERT* | | | |
| *LOCK TABLES* | Позволяет осуществлять блокировки таблиц при помощи операто- ров *LOCK TABLES* и *UNLOCK TABLES*. Для вступления в действие  этой привилегии должна быть установлена привилегия *SELECT* | | | |
| *SELECT* | Позволяет осуществлять выборки таблиц оператором *SELECT* | | | |
| *SHOW DATABASES* | Позволяет просматривать список всех таблиц на сервере при по- мощи  оператора *SHOW DATABASES* | | | |
| *SHOW VIEW* | Позволяет использовать оператор *SHOW CREATE VIEW* | | | |
| *UPDATE* | Позволяет обновлять содержимое таблиц оператором *UPDATE* | | | |
| *USAGE* | Синоним для статуса «отсутствуют привилегии» | | | |
| *GRANT OPTION* | Позволяет управлять привилегиями других пользователей, без данной привилегии невозможно выполнить операторы *GRANT* И  *REVOKE* | | | |

Таблица 17.2. Вместо ключевого слова ON

|  |  |  |
| --- | --- | --- |
| **Ключевое**  **слово *ON*** | | **Уровень** |
| *ON* | *\*.\** | Глобальный уровень – пользователь с полномочиями на глобальном уровне может  обращаться ко всем БД и таблицам, входящим в их состав |
| *ON* | *db.\** | Уровень базы данных – привилегии распространяются на таблицы базы дан-  ных *db* |
| *ON* | *db.tbl* | Уровень таблицы – привилегии распространяются на таблицу *tbl* базы данных  *db* |
| *ON* | *db.tbl* | Уровень столбца – привилегии касаются отдельных столбцов в таблице *tbl*  базы  данных *db*. Список столбцов указывается в скобках через запятую после ключевых слов *SELECT, INSERT, UPDATE* |

### Задание для практической работы

* При выполнении практической работы необходимо:
* создать транзакцию, произвести ее откат и фиксацию;
* составить отчет по практической работе.

### Задание 1. Обработка транзакций

Для выполнения задания объединим несколько операций по добавлению в таблицу *catalogs* новых каталогов, а затем произведем откат транзакции, т. е. отмену произведенных действий. Отключаем режим автоматического завершения, добавляем новые записи и про- веряем, добавились записи или нет.



Откатываем транзакцию оператором *ROLLBACK* (изменения не сохранились).

Воспроизведем транзакцию и сохраним действия оператором *COMMIT*.



### Задание 2. Защита данных

Для работы выберем два компьютера, подключенных к локальной сети. На одном необходимо развернуть сервер MySQL, на другой – скопировать клиент командной строки



*mysql.exe*. Определим IP-адрес сервера:

Создадим новую учетную запись, позволив пользователю *user1* обращаться к серверу MySQL с любых компьютеров сети:



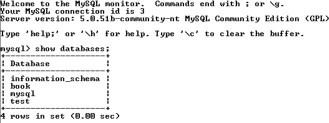
Назначим этому пользователю привилегии глобального уровня:



На клиентском компьютере в командной строке (например, с помощью FAR), запу- стим клиент командной строки в следующем формате:



Наблюдаем отклик удаленного сервера и работаем с ним как обычно:



### Варианты заданий к практическим работам по MySQL

1. *Страховая компания*. Страховая компания имеет филиалы, которые харак- теризуются наименованием, адресом и телефоном. В филиалы обращаются клиенты с це- лью заключения договора о страховании. В зависимости от принимаемых на страхование объектов и страхуемых рисков договор заключается по определенному виду страхования (страхование автотранспорта от угона, страхование домашнего имущества, добровольное медицинское страхование). При заключении договора фиксируются: дата заключения, стра- ховая сумма, вид страхования, тарифная ставка и филиал, в котором заключался договор. Договоры заключают страховые агенты. Помимо информации об агентах (фамилия, имя, отчество, адрес, телефон) нужно хранить филиал, в котором они работают. Необходимо иметь возможность рассчитывать заработную плату агентам. Заработная плата составляет некоторый процент от страхового платежа (платеж – страховая сумма, умноженная на та- рифную ставку). Процент зависит от вида страхования, по которому заключен договор.
2. *Гостиница*. Гостиница предоставляет номера клиентам. Каждый номер ха- рактеризуется вместимостью, комфортностью (люкс, полулюкс, обычный) и ценой. О клиентах собирается

определенная информация (фамилия, имя, отчество, паспортные данные, адрес жительства и некоторый комментарий). Сдача номера клиенту производится при наличии свободных мест в номерах, подходящих клиенту по указанным выше параметрам. При заселении фик- сируется дата заселения. При выезде из гостиницы для каждого места запоминается дата освобождения. Необходимо также осуществлять бронирование номеров. Для постоянных клиентов, а также для определенных категорий клиентов предусмотрена система скидок. Скидки могут суммироваться.

1. *Ломбард*. В ломбард обращаются различные лица с целью получения денеж- ных средств под залог товаров. Клиент сообщает фамилию, имя, отчество и другие паспорт- ные данные. После оценивания стоимости принесенного в качестве залога товара работник ломбарда определяет сумму, которую готов выдать на руки клиенту, а также свои комисси- онные. Кроме того определяется срок возврата денег. Если клиент согласен, то договорен- ности фиксируются в виде документа, деньги выдаются клиенту, а товар остается в лом- барде. Если в указанный срок не происходит возврата денег, товар переходит в собствен- ность ломбарда. После перехода прав собственности на товар, ломбард может продавать товары по цене, меньшей или большей, чем была заявлена при сдаче. Цена может меняться несколько раз, в зависимости от ситуации на рынке (например, владелец ломбарда может устроить распродажу зимних вещей в конце зимы). Помимо текущей цены нужно хранить все возможные значения цены для данного товара.
2. *Оптово-розничная продажа товаров*. Компания торгует товарами из опре- деленного спектра. Каждый товар характеризуется наименованием, оптовой ценой, рознич- ной ценой и справочной информацией. В компанию обращаются покупатели, для каждого из которых в базе данных фиксируются стандартные данные (наименование, адрес, теле- фон, контактное лицо). По каждой сделке составляется документ, в котором наряду с поку- пателем фиксируются количество купленного им товара и дата покупки. Обычно покупа- тели в рамках одной сделки покупают не один товар, а сразу несколько. Также компания

решила предоставлять скидки в зависимости от количества закупленных товаров и их об- щей стоимости.

1. *Ведение заказов*. Компания занимается оптовой продажей различных това- ров. Каждый из товаров характеризуется ценой, справочной информацией и признаком наличия или отсутствия доставки. В компанию обращаются заказчики. Для каждого из них в базе данных запоминаются стандартные данные (наименование, адрес, телефон, контакт- ное лицо). По каждому заказу составляется документ, в котором наряду с заказчиком фик- сируются количество купленного им товара и дата покупки. Доставка товаров может про- изводиться способами, различными по цене и скорости. Нужно хранить информацию о том, какими способами может осуществляться доставка каждого товара, и информацию о том, какой вид доставки (и какую стоимость доставки) выбрал клиент при заключении сделки.
2. *Бюро по трудоустройству*. Бюро готово искать работников для различных работодателей и вакансии для ищущих работу специалистов различного профиля. При об- ращении в бюро работодателя его стандартные данные (название, вид деятельности, адрес, телефон) фиксируются в базе данных. При обращении в бюро соискателя его стандартные данные (фамилия, имя, отчество, квалификация, профессия, иные данные) также фиксиру- ются в базе данных. По каждому факту удовлетворения интересов обеих сторон составля- ется документ. В документе указываются соискатель, работодатель, должность и комисси- онные (доход бюро). В базе должна фиксироваться не только сделка, но и информация по открытым вакансиям. Кроме того для автоматического поиска вариантов необходимо вести справочник «Виды деятельности».
3. *Нотариальная контора*. Нотариальная контора готова предоставить кли- енту определенный комплекс услуг. Услуги формализованы, т. е. составлен их список с описанием каждой услуги. При обращении клиента его стандартные данные (название, вид деятельности, адрес, телефон) фиксируются в базе данных. По каждому факту оказания услуги клиенту составляется документ, в котором указываются дата, услуга, сумма сделки, комиссионные (доход конторы), описание сделки. В рамках одной сделки клиенту может быть оказано несколько услуг. Стоимость каждой услуги фиксирована. Кроме того компа- ния предоставляет в рамках одной сделки различные виды скидок. Скидки могут суммиро- ваться.
4. *Фирма по продаже запчастей*. Фирма продает запасные части для автомоби- лей. Фирма имеет определенный набор поставщиков, по которым известны название, адрес и телефон. У поставщиков приобретаются детали. Каждая деталь характеризуется назва- нием, артикулом и ценой. Некоторые из поставщиков могут поставлять одинаковые детали (один артикул). Каждый факт покупки запчастей у поставщика фиксируется в базе данных, причем обязательными для запоминания являются дата покупки и количество приобре- тенных деталей. Цена детали может

меняться от поставки к поставке. Поставщики заранее ставят фирму в известность о дате изменения цены и ее новом значении. Нужно хранить не только текущее значение цены, но и всю историю изменения цен.

1. *Курсы по повышению квалификации*. В учебном заведении организованы курсы повышения квалификации. Группы слушателей формируются в зависимости от спе- циальности и отделения. В каждую из них включено определенное количество слушателей. Проведение занятий обеспечивает штат преподавателей, для каждого из которых в базе дан- ных зарегистрированы стандартные анкетные данные (фамилия, имя, отчество, телефон) и стаж работы. В результате распределения нагрузки получена информация о том, сколько часов занятий проводит каждый преподаватель с соответствующими группами. Хранятся также сведения о виде занятий (лекция, практика), дисциплине и оплате за 1 час. Размер почасовой оплаты зависит от предмета и типа занятия. Кроме того каждый преподаватель может вести не все предметы, а только некоторые.
2. *Определение факультативов для студентов*. Преподаватели кафедры в

высшем учебном заведении обеспечивают проведение факультативных занятий по некото- рым предметам. Имеются сведения о студентах, включающие стандартные анкетные дан- ные (фамилия, имя, отчество, группа, адрес, телефон). По каждому факультативу суще- ствует определенное количество часов и вид проводимых занятий (лекции, практика, лабо- раторные работы). В результате работы со студентами появляется информация о том, кто из них записался на какие факультативы. Существует некоторый минимальный объем фа- культативных предметов, которые должен прослушать каждый студент. По окончании се- местра в базу данных заносится информация об оценках, полученных студентами на экза- менах. Некоторые из факультативов могут длиться более одного семестра. В каждом се- местре для предмета устанавливается объем лекций, практик и лабораторных работ в часах. В качестве итоговой оценки за предмет берется последняя оценка, полученная студентом.

1. *Распределение учебной нагрузки*. Необходимо распределять нагрузку между преподавателями кафедры. Имеются сведения о преподавателях, включающие наряду с анкетными данными сведения об их ученой степени, занимаемой должности и стаже работы. Преподаватели кафедры должны обеспечить проведение занятий по некото- рым дисциплинам. По каждой из них существует определенное количество часов. В резуль- тате распределения нагрузки необходимо получить информацию следующего рода: «Та- кой-то преподаватель проводит занятия по такой-то дисциплине с такой-то группой». Все проводимые занятия делятся на лекционные и практические. По каждому виду занятий устанавливается свое количество часов. Кроме того данные по нагрузке нужно хранить не- сколько лет.
2. *Распределение дополнительных обязанностей*. Кафедра вуза имеет штат со- трудников, каждый из которых получает определенный оклад. Каждый месяц возникает по- требность в выполнении некоторой дополнительной работы, не входящей в круг основных обязанностей сотрудников. Для наведения порядка в этой сфере классифицированы все виды дополнительных работ и определена сумма оплаты по факту их выполнения. При воз- никновении дополнительной работы назначается ответственный, фиксируется дата начала. По факту окончания фиксируется дата и выплачивается дополнительная сумма к зарплате с учетом классификации. Необходимо учесть разделение сотрудников на преподавателей и учебно-вспомогательный персонал. Для первых нужно хранить сведения об ученой степени и ученом звании, для вторых – о должности. Некоторые работы являются трудоемкими и срочными, что требует привлечения к их выполнению нескольких сотрудников. Длитель- ность работ различна. Нужно заранее планировать длительность работы и количество со- трудников, занятых для выполнения работы.
3. *Техническое обслуживание станков*. Компания занимается ремонтом стан- ков и другого оборудования. Клиентами компании являются промышленные предприятия. Ремонтные работы организованы следующим образом: все станки классифицированы по типам, странам- производителям, годам выпуска и маркам. Все виды ремонта отличаются названием, продолжительностью в днях, стоимостью. Исходя из этих данных, по каждому факту ремонта фиксируется вид станка, дата начала и дата окончания ремонта. Анализ по- казал, что нужно не просто подразделять станки по типам, а иметь информацию о том, сколько раз ремонтировался тот или иной станок.
4. *Туристическая фирма*. Фирма продает путевки клиентам. У каждого кли- ента запрашиваются стандартные данные – фамилия, имя, отчество, адрес, телефон. После этого сотрудники компании выясняют у клиента, куда он хотел бы поехать отдыхать. Ему демонстрируются различные варианты, включающие страну проживания, особенности климата,отель. Обсуждается длительность пребывания и стоимость путевки. Если удалось найти приемлемый вариант, регистрируется факт продажи путевки (или путевок, если кли- ент покупает сразу несколько), фиксируется дата отправления. Иногда клиенту предостав- ляется скидка (скидки фиксированы и могут суммироваться). Фирма работает с несколь- кими отелями (название, категория, адрес) в нескольких странах. Путевки продаются на одну, две или четыре недели. Стоимость путевки зависит от длительности тура и отеля.
5. *Грузовые перевозки*. Компания осуществляет перевозки грузов по различ- ным маршрутам. Необходимо отслеживать стоимость перевозок с учетом заработной платы водителей. Для каждого маршрута определено название, вычислено примерное расстояние и установлена некоторая оплата для водителя. Информация о водителях включает фами- лию, имя, отчество и стаж. Для проведения расчетов хранится полная информация о пере- возках (маршрут, водитель, даты отправки и прибытия). По факту некоторых перевозок во- дителям выплачивается премия. Фирма решила ввести гибкую систему оплаты. Оплата во- дителям должна зависеть не только от маршрута, но и от стажа водителя. Кроме того, нужно учесть, что перевозку могут осуществлять два водителя.
6. *Учет телефонных переговоров*. Телефонная компания предоставляет або- нентам телефонные линии для междугородних переговоров. Абонентами компании явля- ются юридические лица, имеющие телефонную точку, ИНН, расчетный счет в банке. Сто- имость переговоров зависит от города, в который осуществляется звонок, и времени суток (день, ночь). Каждый звонок абонента автоматически фиксируется в базе данных. При этом запоминаются город, дата, длительность разговора и время суток. Компания решила ввести гибкую систему скидок. Так, стоимость минуты теперь уменьшается в зависимости от дли- тельности разговора. Размер скидки для каждого города разный.
7. *Учет внутриофисных расходов*. Сотрудники частной фирмы могут осу- ществлять мелкие покупки для нужд фирмы, предоставляя в бухгалтерию товарный чек. Бухгалтерия отслеживает внутриофисные расходы. Фирма состоит из отделов, каждый из которых имеет название. В каждом отделе работает определенное количество сотрудников. Сотрудники могут осуществлять покупки в соответствии с видами расходов. Каждый вид расходов имеет название, некоторое описание и предельную сумму средств, которые могут быть потрачены по данному виду расходов в месяц. При каждой покупке сотрудник оформ- ляет документ, где указывает вид расхода, дату, сумму и отдел. Нужно хранить данные о расходах не только в целом по отделу, но и по отдельным сотрудникам. Нормативы по рас- ходованию средств устанавливаются не в целом, а по каждому отделу за каждый месяц. Неиспользованные в текущем месяце деньги могут быть использованы позже.
8. *Библиотека*. Библиотека решила зарабатывать деньги, выдавая напрокат книги, имеющиеся в небольшом количестве экземпляров. У каждой книги, выдаваемой в прокат, есть название, автор, жанр. В зависимости от ценности книги для каждой из них определена залоговая стоимость (сумма, вносимая клиентом при взятии книги напрокат) и стоимость проката (сумма, которую клиент платит при возврате книги, получая назад за- лог). Читатели регистрируются в картотеке, которая содержит стандартные анкетные дан- ные (фамилия, имя, отчество, адрес, телефон). Каждый читатель может обращаться в биб- лиотеку несколько раз. Все обращения читателей фиксируются, при этом по каждому факту выдачи книги запоминаются дата выдачи и ожидаемая дата возврата. Стоимость проката книги должна зависеть не только от самой книги, но и от срока ее проката. Кроме того, необходимо добавить систему штрафов за вред, нанесенный книге, и систему скидок для некоторых категорий читателей.
9. *Прокат автомобилей*. Фирма, занимающаяся прокатом автомобилей, имеет автопарк, содержащий некоторое количество автомобилей различных марок, стоимостей и типов. Каждый автомобиль имеет свою стоимость проката. В пункт проката обращаются клиенты. Клиенты проходят обязательную регистрацию, в ходе которой о них собирается стандартная информация (фамилия, имя, отчество, адрес, телефон). Каждый клиент может обращаться в пункт проката несколько раз. Обращения клиентов фиксируются, при этом по каждой сделке запоминаются дата выдачи и ожидаемая дата возврата. Стоимость проката автомобиля должна зависеть не только от самого автомобиля, но и от срока его проката, а также от года выпуска. Также нужно ввести систему штрафов за возвращение автомобиля в ненадлежащем виде и систему скидок для постоянных клиентов.
10. *Выдача банком кредитов*. Коммерческий банк выдает кредиты юридиче- ским лицам. В зависимости от условий получения кредита, процентной ставки и срока

возврата все кредитные операции делятся на несколько основных видов. Каждый из этих видов имеет свое название. Кредит может получить юридическое лицо (клиент), при реги- страции предоставивший следующие сведения: название, вид собственности, адрес, теле- фон, контактное лицо. Каждый факт выдачи кредита регистрируется банком, при этом фик- сируются сумма кредита, клиент и дата выдачи. Чтобы отслеживать динамику возврата кре- дитов принято решение учитывать в системе еще и дату фактического возврата денег. Нужно еще учесть, что кредит может гаситься частями, и за задержку возврата кредита начисляются штрафы.

1. *Инвестиционная компания*. Компания занимается вложением денежных средств в ценные бумаги, которые характеризуются рейтингом, доходностью за прошлый год, минимальной суммой сделки и некоторой дополнительной информацией. Клиентами компании являются предприятия, которые доверяют ей управлять их свободными денеж- ными средства на определенный период. Необходимо выбрать вид ценных бумаг, которые позволят получить прибыль и компании и клиенту. При работе с клиентом существенной является информация о предприятии – название, вид собственности, адрес и телефон. Каж- дая инвестиция характеризуется информацией о клиенте, информацией о ценной бумаге, котировкой бумаги, датой ее покупки и датой ее продажи. Необходимо хранить историю котировок каждой ценной бумаги. Кроме того, помимо вложений в ценные бумаги суще- ствует возможность вкладывать деньги в банковские депозиты.
2. *Занятость актеров театра*. Коммерческий директор театра организует привлечение актеров и заключение контрактов. Каждый год театр осуществляет постановку различных спектаклей. Каждый спектакль имеет определенный бюджет. Для участия в кон- кретных постановках в определенных ролях привлекаются актеры. С каждым из актеров заключается персональный контракт на определенную сумму. Каждый актер имеет стаж работы, некоторые из них удостоены различных званий. В рамках одного спектакля на одну и ту же роль привлекается несколько актеров. Контракт определяет базовую зарплату ак- тера, а по итогам реально отыгранных спектаклей актеру назначается премия. В базе дан- ных нужно хранить информацию за несколько лет.
3. *Платная поликлиника*. В поликлинике работают врачи различных специ- альностей, имеющие разную квалификацию. Каждый день в поликлинику обращаются па- циенты. Все пациенты проходят обязательную регистрацию, при которой в базу данных заносятся стандартные анкетные данные (фамилия, имя, отчество, год рождения, адрес). При обращении в поликлинику пациент обследуется и проходит лечение у разных специа- листов. Каждый пациент может обращаться в поликлинику несколько раз, нуждаясь в раз- личной медицинской помощи. Все обращения пациентов фиксируются, при этом устанав- ливается диагноз, определяется стоимость лечения, запоминается дата обращения. Общая стоимость лечения зависит от стоимости консультаций и процедур, назначенных пациенту. Для определенных категорий граждан предусмотрены скидки.
4. *Анализ динамики показателей финансовой отчетности предприятий*. Информационно-аналитический центр крупного холдинга отслеживает динамику показате- лей предприятий холдинга. В структуру холдинга входят несколько предприятий. Каждое предприятие имеет стандартные характеристики (название, реквизиты, телефон, контакт- ное лицо). Работа предприятия может быть оценена следующим образом: в начале каждого отчетного периода на основе финансовой отчетности вычисляется определенный набор по- казателей. Важность показателей характеризуется некоторыми числовыми константами. Значение каждого показателя измеряется в некоторой системе единиц. Некоторые показа- тели считаются в рублях, некоторые в долларах, некоторые в евро. Для удобства работы с показателями нужно хранить изменения курсов валют относительно друг друга.
5. *Учет телекомпанией стоимости прошедшей в эфире рекламы*. Работа ком- мерческой службы телевизионной компании построена следующим образом: заказчики просят поместить свою рекламу в определенной передаче в определенный день. Каждый

рекламный ролик имеет определенную продолжительность. Для каждой организации-за- казчика известны банковские реквизиты, телефон и контактное лицо для проведения пере- говоров. Передачи имеют определенный рейтинг. Стоимость минуты рекламы в каждой пе- редаче определяется, исходя из рейтинга передачи и прочих соображений. Необходимо хра- нить информацию об агентах, заключивших договоры на рекламу. Зарплата рекламных агентов составляет некоторый процент от общей стоимости рекламы, прошедшей в эфире.*IT-компания*. Компания оказывает IT-услуги организациям и предприятиям. В ком- пании работают сотрудники, о которых должна сохраняться стандартная информация и данные о квалификации (владение языками и системами программирования, знание СУБД, операционных систем). В компанию обращаются клиенты, о которых собираются стандарт- ные данные (наименование и адрес организации, телефон, адрес электронной почты, фами- лия, имя и отчество контактного лица для связи). Задание для клиента выполняет опреде- ленный сотрудник, при этом фиксируется дата выдачи задания и трудоемкость выполнения (в часах). При повторном обращении клиент переходит в категорию постоянных и получает скидку. С ростом компании возникла необходимость разделения ее на отделы. Увеличились масштабы проектов, и теперь задание клиента поручается отделу. В рамках одного договора может выполняться несколько заданий разными отделами компании.

1. *Ювелирная мастерская*. Ювелирная мастерская осуществляет изготовление ювелирных изделий для частных лиц на заказ. Мастерская работает с определенными ма- териалами (платина, золото, серебро, драгоценные камни). При обращении потенциального клиента выясняется, какое именно изделие ему необходимо. Все изготавливаемые изделия принадлежат к некоторому типу (серьги, кольца, броши, браслеты), выполняются из опре- деленного материала, имеют некоторый вес и цену (включающую стоимость материалов и работы). Ювелирное изделие может состоять из нескольких материалов. Кроме того, посто- янным клиентам мастерская предоставляет скидки.
2. *Парикмахерская*. Парикмахерская стрижет клиентов в соответствии с их по- желаниями и некоторым каталогом различных видов стрижки. Для каждой стрижки опре- делены название, категория (мужская, женская, детская), стоимость работы. Для наведения порядка составляется база данных клиентов, где запоминаются их анкетные данные (фами- лия, имя, отчество). Начиная с 5-ой стрижки, клиент переходит в категорию постоянных и получает скидку в 3 % при каждой последующей стрижке. После того, как закончена оче- редная работа, в БД фиксируются стрижка, клиент и дата производства работ. Кроме того, у парикмахерской появился филиал и необходима раздельная статистика по филиалам. Сто- имость стрижки может меняться с течением времени. Нужно хранить не только последнюю цену, но и все данные по изменению цены стрижки.
3. *Химчистка*. Химчистка осуществляет прием у населения вещей для выведе- ния пятен. Для наведения порядка составляется база данных клиентов, в которой запоми- наются их анкетные данные (фамилия, имя, отчество, адрес, телефон). Начиная с 3-го обра- щения, клиент переходит в категорию постоянных клиентов и получает скидку в 3 % при чистке каждой последующей вещи. Все оказываемые услуги подразделяются на виды, име- ющие название, тип и стоимость, зависящую от сложности работ. Работа с клиентом пер- воначально состоит в определении объема работ, вида услуги и, соответственно, ее стоимо- сти. Если клиент согласен, он оставляет вещь (при этом фиксируется услуга, клиент и дата приема) и забирает ее после обработки (при этом фиксируется дата возврата). Химчистка заключает с клиентом договор. Клиент может одновременно сдавать в чистку несколько вещей. У химчистки появились филиалы, и необходима раздельная статистика по филиа- лам. Введены надбавки за срочность и сложность.
4. *Сдача в аренду торговых площадей*. Торговый центр сдает в аренду коммер- сантам свои торговые площади. В результате планирования определено некоторое количе- ство торговых точек в пределах здания, которые могут сдаваться в аренду. Для каждой из торговых точек важными данными являются этаж, площадь, наличие кондиционера и сто-

имость аренды в день. С потенциальных клиентов собираются стандартные данные (назва- ние, адрес, телефон, реквизиты, контактное лицо). При появлении потенциального клиента ему показывают имеющиеся свободные площади. При достижении соглашения оформля- ется договор и в базе данных фиксируется торговая точка, клиент, период (срок) аренды. Некоторые клиенты в рамках одного договора арендуют сразу несколько торговых точек, причем для каждой точки возможен свой срок аренды. Дата заключения договора может не совпадать с датой начала аренды. Необходимо собирать информацию об ежемесячных пла- тежах, поступающих от арендаторов.

## Ркомендуемые источники финансирования Основная литература:

1. Коломейченко, А. С. Информационные технологии : учебное пособие для СПО

/ А. С. Коломейченко, Н. В. Польшакова, О. В. Чеха. - 2-е изд., перераб. - Санкт-Петербург

: Лань, 2021. - 212 с. - URL: https://e.lanbook.com/book/177031 (дата обращения: 25.08.2022).

* Режим доступа: ЭБС Лань, для зарегистрир. пользователей - ISBN 978-5-8114-7565-0. - Текст : электронный.

1. Голицына, О. Л. Информационные системы и технологии : учебное пособие / О. Л. Голицына, Н. В. Максимов, И. И. Попов. - Москва : ФОРУМ : ИНФРА-М, 2021. - 400 с.

* (Среднее профессиональное образование). - URL: https://znanium.com/catalog/product/1138895 (дата обращения: 19.03.2022). - Режим доступа: ЭБС Znanium.com, для зарегистрир. пользователей. - ISBN 978-5-00091-592-9. - Текст : электронный.

1. Информационные технологии : учебное пособие / Л. Г. Гагарина, Я. О. Теплова, Е. Л. Румянцева, А. М. Баин ; под ред. Л. Г. Гагариной. - Москва : ФОРУМ : ИНФРА-М, 2019. - 320 с. - (Профессиональное образование). - URL: https://znanium.com/catalog/product/1018534 (дата обращения: 25.08.2022). - Режим доступа: ЭБС Znanium.com, для зарегистрир. пользователей. - ISBN 978-5-8199-0608-8. - Текст : электронный.
2. Гаврилов, М. В. Информатика и информационные технологии : учебник для среднего профессионального образования / М. В. Гаврилов, В. А. Климов. - 4-е изд., перераб. и доп. – Москва : Юрайт, 2022. - 383 с. - (Профессиональное образование). - URL: https://ezpro.fa.ru:3217/bcode/489603 (дата обращения: 17.03.2022). - Режим доступа: ЭБС Юрайт, для зарегистрир. пользователей. - ISBN 978-5-534-03051-8. - Текст : электронный.

## Дополнительная литература:

1. Федотова, Е. Л. Информационные технологии в профессиональной деятельности : учебное пособие / Е. Л. Федотова. - Москва : ФОРУМ : ИНФРА-М, 2021. -

367 с. - (Среднее профессиональное образование). - URL: https://znanium.com/catalog/product/1189329 (дата обращения: 19.03.2022). - Режим доступа: ЭБС Znanium.com, для зарегистрир. пользователей. - ISBN 978-5-8199-0752-8. - Текст : электронный.

1. Гвоздева, В. А. Информатика, автоматизированные информационные технологии и системы : учебник / В. А. Гвоздева. - Москва : ФОРУМ : ИНФРА-М, 2021. -

542 с. - (Среднее профессиональное образование). - URL: https://znanium.com/catalog/product/1190684 (дата обращения: 19.03.2022). - Режим доступа: ЭБС Znanium.com, для зарегистрир. пользователей. - ISBN 978-5-8199-0856-3. - Текст : электронный.

1. Синаторов, С. В. Информационные технологии. Задачник : учебное пособие / Синаторов С. В. - Москва : КноРус, 2022. - 253 с. - URL: https://book.ru/book/934646 (дата обращения: 19.03.2022). - Режим доступа: ЭБС Book.ru, для зарегистрир. пользователей. - ISBN 978-5-406-09306-1. - Текст : электронный.
2. Советов, Б. Я. Информационные технологии : учебник для среднего профессионального образования / Б. Я. Советов, В. В. Цехановский. - 7-е изд., перераб. и доп. - Москва : Юрайт, 2022. - 327 с. - (Профессиональное образование). - URL: <https://ezpro.fa.ru:3217/bcode/489604> (дата обращения: 17.03.2022). - Режим доступа: ЭБС Юрайт, для зарегистрир. пользователей. - ISBN 978-5-534-06399-8. - Текст : электронный.

## Электронные ресурсы:

1. [http://www.ed.gov.ru](http://www.ed.gov.ru/) – Министерство образования Российской федерации.
2. [http://www.edu.ru](http://www.edu.ru/) – Федеральный портал «Российское образование».
3. [http://www.yandex.ru](http://www.yandex.ru/) – Русская поисковая система.
4. <http://ict.edu.ru/lib/> - Библиотека портала «ИКТ в образовании»
5. <http://window.edu.ru/-> Единое окно доступа к образовательным ресурсам
6. <http://www.firo.ru/> - Министерство образования и науки РФ ФГАУ «ФИРО»
7. [http://www.consultant.ru](http://www.consultant.ru/). - Справочно-правовая система «Консультант Плюс»
8. [http://www.garant.ru](http://www.garant.ru/) - Справочно-правовая система «Гарант».
9. [http://www.minfin.ru](http://www.minfin.ru/). – Официальный сайт Министерства финансов Российской Федерации
10. [http://www.nalog.ru.](http://www.nalog.ru/) - Официальный сайт Федеральной налоговой службы
11. [http://znanium.com](http://znanium.com/) – Электронно-библиотечная система znanium.com
12. [http://www.urait.ru](http://www.urait.ru/) – электронная библиотека издательства ЮРАЙТ