Министерство науки и высшего образования Российской Федерации

Южно-Уральский государственный университет

Институт естественных и точных наук

Факультет математики, механики и компьютерных технологий

Кафедра прикладной математики и программирования

004.6(07)

О535

**АДМИНИСТРИРОВАНИЕ И ПРОЕКТИРОВАНИЕ   
ХРАНИЛИЩ ДАННЫХ**

Методические указания по выполнению курсовых работ

Челябинск

Издательский центр ЮУрГУ

2019

УДК 004.658(075.8)

О535

Одобрено

учебно-методической комиссией института естественных и точных наук

Рецензент

Н.В. Плотникова

|  |  |
| --- | --- |
|  | Оленчикова, Т.Ю. |
| О535 | **Администрирование и проектирование хранилищ данных:** методические указания по выполнению курсовых работ / Т.Ю. Оленчикова, М.Ю.Сартасова. – Челябинск: Издательский центр ЮУрГУ, 2019. – 56 с. |
|  | В методическом издании рассматриваются этапы разработки курсовой работы по дисциплине «Администрирование и проектирование хранилищ данных», приводятся критерии оценки, требования к оформлению и содержанию пояснительной записки и презентации.  Работа состоит из двух частей: проектирования реляционной базы данных и разработки клиент-серверного приложения. Представлены этапы разработки базы данных от концептуального проектирования до реализации в конкретной СУБД MS SQL Server. Особое внимание уделено механизмам обеспечения целостности и конфиденциальности данных на стороне сервера. Даны рекомендации по выполнению системного анализа предметной области и документированию принятых решений. Обсуждаются особенности взаимодействия клиентского приложения с базой данных по технологии ADO.NET.В приложениях приводятся примеры оформления основных частей пояснительной записки,  Методическое издание предназначено для студентов направлений 01.03.02 «Прикладная математика и информатика», 01.03.04 «Прикладная математика». |

УДК 004.658(075.8)

# ЭТАПЫ ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ

Курсовая работа (КР) по дисциплине «Администрирование и проектирование хранилищ данных» представляет собой учебно-практическое исследование, предназначенное для систематизации, углубления и закрепления знаний, полученных студентом в процессе изучения данной дисциплины в соответствии с учебным планом.

Цель курсовой работы – приобретение студентом практических навыков по разработке, формулированию требований, построению моделей и ведению баз данных (БД), их отладке и документированию, а также по разработке информационных систем. Для реализации данной цели студент должен:

* уметь проводить системный анализ объекта автоматизации и на его основе формулировать требования к разрабатываемой информационной системе (ИС);
* владеть технологией проектирования БД;
* иметь навыки администрирования и ведения БД;
* уметь распределять задачи между клиентским приложением и сервером баз данных;
* знать основы обеспечения целостности и безопасности данных в многопользовательской среде, разграничения прав доступа пользователей;
* уметь создавать информационные системы для решения задач автоматизации;
* показать умение применять теоретические положения к решению практических задач, четко формулировать свои мысли и предложения.

Курсовая работа ориентирована на разработку структуры реляционной базы данных и создание на ее основе автоматизированной информационной системы, предназначенной для решения конкретных прикладных задач. Проект выполняется в среде MS Visual Studio на языке программирования C#, в качестве СУБД используется MS SQL Server.

Перечень этапов, распределение времени и баллов приведены в таблице 1.

Таблица 1 – Продолжительность и трудоемкость этапов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование этапа | Продолжи-тельность | Часы | Баллы | Результат работы |
| 1 | 2 | 3 | 4 | 5 |
| Разработка технического задания | 2 недели | 4 | 10 | Техническое задание на курсовое проектирование |
| Концептуальное (инфологическое) проектирование БД | 2 недели | 4 | 15 | ER-модель БД, бизнес-процессы, типы и ограничения данных |
| Логическое проектирование БД | 1 неделя | 4 | 10 | Схема реляционной БД, нормализация |

Таблица 1 – Продолжение

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| Разработка объектов БД на стороне сервера (физическое проектирование | 2 недели | 4 | 15 | Создание необходимых объектов БД в MS SQL Server: таблицы, связи, триггеры, индексы, функции, хранимые процедуры |
| Разработка интерфейса клиентского приложения | 1 неделя | 4 | 10 | Экранные формы |
| Создание, отладка и тестирование клиентского приложения | 3 недели | 8 | 20 | Исходный текст |
| Оформление программной документации и отчета по курсовой работе | 2 недели | 4 | 10 | Пояснительная записка |
| Защита курсовой работы | 1 неделя | 4 | 10 | Презентация к докладу |
| ВСЕГО | 16 недель | 36 | 100 |  |

Оценка за курсовую работу выставляется по числу набранных баллов согласно «Положению о балльно-рейтинговой системе ЮУрГУ» (БРС) [1].

# ТЕМЫ ДЛЯ КУРСОВЫХ РАБОТ

Для каждой темы из списка приведен краткий перечень объектов, функций и отчетов, которые необходимо подготовить во время выполнения курсовой работы.

1) Система для продажи авиабилетов

Объекты: авиакомпании, аэропорты, типы самолетов (мест), экземпляры самолетов, расписание, пассажиры (ф.и.о, паспорт, № билета), цены.

Основная бизнес-функция: продать билет, подбор рейсов по маршруту.

Отчеты: списки пассажиров по каждому рейсу, расписание рейсов.

2) Система для продажи железнодорожных билетов

Железные дороги, станции, расписание, цены, маршруты (поезд №), экземпляры поездов, типы вагонов, экземпляры вагонов, проданные билеты

Основная бизнес-функция: продать билет, подбор рейсов по маршруту.

Отчеты: средняя загрузка вагонов за заданный период с группировкой по типам вагонов и направлениям; число перевезенных пассажиров; расписание рейсов.

3) Справочник по лекарственным растениям, сборам

Название растения (русский, латынь), лекарственные сборы, заболевания, лекарственные формы (отвары, настои, порошки:), способ применения.

Основная бизнес-функция: поиск лекарственных сборов для лечения заболевания

Отчеты: подборка лекарственных сборов для лечения заболевания с группировкой по лекарственным формам и с учетом противопоказаний.

4) Склад

Некоторая фирма имеет склады (№ склада, адрес), на складе работают кладовщики, которые принимают и отпускают товары. Товары поступают на склад по накладным (№,дата, от кого), в накладной для каждого товара указано количество и цена. Товары могут также поступать и от собственных подразделений производства, которые эти товары производят. В этом случае они также сопровождаются накладной. Товары продаются фирмой внешним организациям по цене назначаемой фирмой. Цены продажи в каждый данный момент фиксированы и отражаются в прейскуранте (pricelist). Товары также передаются по накладной собственным подразделениям для использования в процессе производства.

Основная бизнес-функция: учет наличия и перемещения товаров.

Отчеты: отпущенные, принятые товары за заданный промежуток времени с группировкой по товару или по поставщикам/покупателям: наименование товара, откуда, кому, количество, цена,…

5) Расчеты за коммунальные услуги

Домовладельцы, дома, коммунальные услуги (холодная вода, горячая вода, газ, электроплиты). Коммунальные услуги имеют цену, которая исчисляется либо по числу жильцов, либо по квадратным метрам общей площади. Поставщики услуг. Клиенты (жильцы) проживают в квартирах. Клиенты вносят плату за жилье и услуги.

Основная бизнес-функция: учет начисления и оплаты услуг.

Отчеты: задолженность жильцов по оплате на некоторую дату с указанием вида услуг.

6) Метрологическая служба предприятия

Предприятие имеет ряд подразделений, в которых используются измерительные приборы. Метрологическая служба следит за состоянием приборов. Прибор имеет вид (амперметр) и тип (конкретное наименование модели, например, Е-12U6). Для типа прибора определена периодичность поверки (1 раз в 6 месяцев). Для каждого экземпляра прибора хранится дата последней поверки. Прибор имеет конкретного производителя и гарантийный срок, назначенный производителем для этого типа. Существует некоторое множество характеристик приборов (ток, напряжение, размеры …). Для каждой характеристики существует множество возможных значений. Тип прибора может обладать некоторой характеристикой, например, ток 5 ампер.

Основная бизнес-функция: учет приборов: характеристики, дата поверки, срок годности.

Отчеты: показать список приборов, годных для измерений, по условиям поверки, для выполнения измерений заданных физических величин; показать список приборов, у которых заканчивается или закончился срок поверки; показать список неисправных приборов.

7) Предприятие автосервиса

Предприятие автосервиса располагает цехами. В цехах работают мастера, каждый из которых выполняет работу некоторого вида (малярные, электротехнические:). Клиент сдает машину в ремонт, при этом оформляется заказ, содержащий перечень работ. Работа относится к некоторому виду и для своего выполнения требует определенного количества материалов и комплектующих изделий, а также времени. Имеются расценки на материалы и комплектующие. Конкретная работа из заказа выполняется мастером. Мастер не должен быть загружен более 8 часов в сутки.

Основные бизнес-функции: учет заказов, планирование работ.

Отчеты: расчет стоимости заказа, скидки постоянным заказчикам.

8) Междугородние автобусные перевозки

Задана география автомобильных дорог. Некоторое множество АТП имеет парки автобусов. Подлежит исполнению множество рейсов, имеющих определенную регулярность. Рейс передвигается по дорогам, делая остановки в пунктах. Известны цены и время перемещения марки автобуса между пунктами. Автобусы ведут водители. Продажа билетов фиксируется в БД.

Основная бизнес-функция: продать билет.

Отчеты: списки автобусов, находящихся на маршрутах, где находится в данное время; число перевезенных пассажиров за заданный период, подбор рейсов по маршруту.

9) Грузовые морские перевозки

Компания обладает некоторым флотом и берет подряды на морские контейнерные перевозки. Она может также фрахтовать суда других судовладельцев. Судно плавает под флагом некоторой страны. Контейнер (№ контейнера) содержит грузы одного клиента. Грузы одного клиента описаны в накладной, содержащей перечень контейнеров и содержащихся в них объектов (количество), с разными единицами измерения объема. Судно в конкретном рейсе имеет определенный маршрут, задаваемый последовательностью портов, датами прибытия и отплытия. В каждом порту происходит погрузка и выгрузка контейнеров. Грузы сопровождают агенты компании (материально ответственные лица).

Основная бизнес-функция: учет перемещения грузов.

Отчеты: по номеру квитанции определить настоящее месторасположения груза, куда он движется.

10) Диспетчер троллейбусного парка

Троллейбусный парк располагает некоторым множеством машин (троллейбусов), имеющих номер и дату производства. Каждая машина может находиться в состоянии: работа, неисправность, капитальный ремонт, производимый с некоторой периодичностью. Парк обслуживает некоторое множество маршрутов. Маршрут имеет время прохождения. Водители водят машины и имеют категорию. Работа водителя определенной категории оплачивается по некоторому тарифу. Парк работает в три смены. Функции диспетчера заключаются в назначении машин и водителей для выполнения маршрута. Планирование ведется на неделю вперед. Хранится некоторая история выполненных работ, которую использует бухгалтерия для начисления зарплаты.

Основная бизнес-функция: назначение водителей на маршруты с учетом ограничений на длительность рабочего времени и др.

Отчеты: время начала, окончания работы, номер маршрута, троллейбуса по каждому водителю.

11) Учебный процесс в школе

Программа для завуча, имеющая целью распределение нагрузки и составление расписания для выполнения учебного плана. Расписание составляется на каждую неделю. Имеется фонд аудиторий, их характеристики. Имеется учебный план и преподаватели, задействованные в учебном процессе.

Основная бизнес-функция: заполнение расписания с проверкой ограничений целостности и непротиворечивости.

Отчеты: расписание преподавателя, расписание класса на заданный период, распределение нагрузки по преподавателям.

12) Журнал "Пульс цен"

Фирмы торгуют компьютерами и комплектующими. Комплектующие (материнская плата, процессор и т.п.) имеют цену, назначенную фирмой. Существуют стандартные конфигурации компьютеров, предлагаемые фирмами, состав которых предопределен. Комплектующие имеют производителя и характеристики. Характеристики способны принимать значения.

Основные бизнес-функции: учет комплектующих и компьютеров, продажа, сборка компьютеров из комплектующих.

Отчеты: поиск товара по цене, типу, характеристикам, производителям.

13) Кафедра

Кафедра выпускает ряд специальностей. На кафедре работают преподаватели и вспомогательный персонал. Студенческие группы относятся к специальности. Преподаватели читают курсы и выполняют другие виды нагрузки. Учебные планы специальности предусматривают определенное число часов по виду занятия для курса в семестре. Группы могут объединяться в потоки. Студенты сдают экзамены, зачеты и курсовые работы. Есть нормативы по расчету нагрузки.

Основная бизнес-функция: распределение нагрузки по преподавателям.

Отчеты: нагрузка преподавателя на заданный семестр.

14) Строительные ремонтные работы

Организация выполняет строительные ремонтные работы по договорам (№ договора, дата договора). Работы выполняются над объектами (код, название, адрес). Работы подразделяются на виды (каменные, земляные, малярные), имеют единицу измерения и тариф на выполнение единицы работы. В рамках договора выполняется некоторое количество некоторых видов работ. Для выполнения работ необходимы рабочие (специальность, разряд, тарифная ставка); техника (группа, марка); материалы (код, название, единица измерения, цена). По мере выполнения работ закрываются акты приемки-сдачи выполненных работ. На основании акта формируется счет на оплату работ, который предъявляется заказчику, и который его оплачивает платежным поручением через свой банк.

Основная бизнес-функция: учет выполненных работ, их оплаты.

Отчеты: расчет стоимости работ согласно установленным тарифам.

15) Железнодорожные грузовые перевозки

Железная дорога имеет с организациями договоры на перевозку грузов. Груз сопровождается накладной (номер, дата, организация). В накладной перечислены грузы и их количество. Товары, перевозимые по накладной, едут в одном вагоне. Вагон имеет номер, емкость и вид (платформа, цистерна). Груз отправляется одной организацией и будет получен другой (возможно той же самой). Груз едет от пункта отправления до пункта назначения. Вагон прицепляется к поезду, имеющему маршрут, номер и дату. В процессе движения он может быть перецеплен к другому поезду.

Основная бизнес-функция: учет перемещения груза.

Отчеты: поиск груза по номеру накладной, где находится или куда отправлен.

16) Конный спорт

Лошади и их родословные, жокеи, владельцы лошадей (лошадь может быть продана), тренеры, конюшни, принадлежащие владельцам, состязания, их виды, ставки в тотализаторе (например, выплата 1:20).

Основная бизнес-функция: учет родословной и достижений лошадей.

Отчеты: «Достижения (победы)» каждой лошади; «Досье (характеристика)» лошади: возраст, тренер, конюшня, владелец и т.п.

17) Отдел кадров

Предприятие имеет определенную номенклатуру должностей (код, наименование, тариф). Предприятие делится на подразделения, подразделения на отделы. Каждое подразделение или отдел имеют руководителя. Каждое подразделение или отдел имеют штатное расписание. Должности по штатному расписанию занимают сотрудники. Сотрудник имеет атрибуты (Ф.И.О., год рождения). Сотрудник имеет послужной список. Сотрудник имеет поощрения и взыскания. Сотрудник имеет некоторую историю получения образования и повышения квалификации.

Основная бизнес-функция: учет сотрудников

Отчеты: послужной список сотрудника: должности, награды, поощрения; штатное расписание подразделения на указанную дату с отметкой незанятых штатных единиц.

18) Частный предприниматель

Частный предприниматель - программист выполняет договорные работы на разработку и сопровождение программного обеспечения. Его заказчиками являются организации. Договоры разбиваются на этапы, каждый этап имеет цену. Для выполнения работ необходимы затраты на комплектующие компьютера, программное обеспечение, литературу. Если объем договора слишком велик, то предприниматель заключает договоры субподряда со своими коллегами. Все расчеты по договорам выполняются платежными поручениями через банки ЧП и заказчиков (или субподрядчиков).

Основная бизнес-функция: учет выполнения договоров, субподрядчиков.

Отчеты: выполненные и принятые этапы договоров и субподрядов, их оплата.

19) Выплавка стали

Завод получает заказы на выплавку стали определенных марок в определенном количестве. Завод располагает сталеплавильными печами. Печь относится к некоторому типу, и за одну плавку позволяет получить определенное количество стали. Для марки стали и типа печи имеется график энергопотребления в процессе плавки, задаваемый таблицей (время от начала плавки, киловатт). Стоимость электроэнергии меняется в течение суток и задана таблицей (час суток, тариф). Программа оптимизирует распределение плавок по печам для выполнения заказов за некоторый период (месяц или неделя). Результатом является план в виде отношения (№ заказа, марка стали, день и время начала плавки). Это отношение затем может корректироваться вручную.

Основная бизнес-функция: планирование процесса выплавки стали.

Отчеты: объемы выплавляемой стали и их марки, расход электроэнергии на выплавку.

20) Туризм

Фирма специализируется по организации спортивных водных туристических походов по рекам России. Существует множество маршрутов. Маршруты имеют категории сложности. Маршрут может представлять собой несколько отрезков перемещения (начало, конец, расстояние, способ перемещения – пеший или сплав, дней). На участках сплава имеются пороги, имеющие собственные имена. Пороги классифицируются по сложности. Фирма располагает штатом инструкторов, аттестованных на проведение походов заданной категории сложности. Инструкторы оплачиваются по тарифной ставке (за день).

Фирма располагает снаряжением, относящимся к некоторой группе (палатки, плоты, катамараны, спальные мешки). Каждое изделие группы относится к типу. Тип – множество одинаковых изделий. Тип имеет цену и ресурс, выражаемый в числе походов. Изделие типа имеет инвентарный номер.

Фирма набирает группу туристов, которые оплачивают услуги фирмы (экземпляр похода). Стоимость похода складывается из затрат на дорогу, питание, амортизацию снаряжения, оплаты инструкторов и фиксированный процент на доход фирмы.

Основные бизнес-функции: планирование маршрутов, учет использования снаряжения.

Отчеты: стоимость, длительность маршрута.

21) Охотничье хозяйство

Охотничье хозяйство разбито на участки. На участках водится живность разных видов в определенном количестве на квадратный километр. Участок может быть сдан для охоты группе охотников на некоторый отрезок времени. В лицензии для группы указывается, какое количество и какого зверя они имеют право добыть, а также цена за единицу вида. Учтите, что число оставшихся на участке животных после охоты не должно быть меньше заданного минимального значения, лицензия не может быть предоставлена двум разным группам на один и тот же участок в один и тот же отрезок времени. После охоты учитывается количество добытых зверей.

Основная бизнес-функция: планирование и выдача лицензий.

Отчеты: количество и виды живности по участкам, стоимость лицензии.

22) Гостиница

Гостиница содержит номера, для которых указывается: класс номера, максимальное число проживающих, стоимость проживания 1 гостя по выходным и будням. Ведется учет гостей гостиницы (Ф.И.О., паспорт, дата заезда, предполагаемая дата отъезда, предоплата). Возможно бронирование номеров в гостинице, за которое взимается дополнительная плата от ½ стоимости номера в сутки.

Основная бизнес-функция: учет проживающих в номерах, сдача мест в гостинице гостям.

Отчеты: наличие свободных мест в гостинице и их характеристика, расчеты по оплате за проживание с учетом брони.

23) Электронный журнал студенческой группы

Группа студентов обучается по общему учебному плану, в котором указаны предметы, преподаватели, типы занятий (лекция, лабораторная работа, практическое занятие, контрольные мероприятия) и количество часов занятий. На занятиях преподаватель выставляет оценки студентам в журнал, по журналу формируется итоговая оценка за предмет. Неудовлетворительно или неявка на контрольные мероприятия является основанием считать, что предмет учащимся не освоен. Преподаватель может настроить коэффициенты журнала, чтобы учитывать важность каждого мероприятия, выделить в отдельную категорию темы (модули) дисциплины.

Основная бизнес-функция: учет успеваемости учащихся, настройка журнала

Отчеты: успеваемость по предмету, по всем предметам курса, должники по предмету на заданную дату.

24) Семейный бюджет

Учет доходов и расходов семьи, анализ статей расходов. Под доходными статьями можно понимать любые источники средств. Например, зарплата, стипендия, подработки, финансовая помощь от родителей и тому подобное.

Расходы также можно разделить на множество подразделов: траты на еду; хозяйственные нужды; расходы на транспорт; коммунальные услуги. Пользователь может сам вводить нужные ему статьи и подразделы доходов и расходов.

Основная бизнес-функция: учет и анализ статей доходов и расходов семьи.

Отчеты: доходы и расходы за указанный период с возможностью группировки и сортировки по указанным полям.

25) Больница

Каждая больница города состоит из одного или нескольких корпусов. В корпусе размещается одно или несколько отделений. Отделение специализируется на лечении определенной группы заболеваний. Каждое отделение  имеет некоторое количество палат на определенное число коек. У больницы имеются нормативы на длительность лечения (нахождения пациента в стационаре). Для каждого больного ведется история болезни, которую заполняет лечащий врач. В истории болезни указываются назначенные процедуры и их влияние на состояние больного. Для уточнения диагноза пациент может быть направлен на консультацию в другое отделение.

Кроме того, база данных должна содержать сведения для расчета стоимости лечения: стоимость нахождения пациента в больнице, стоимость назначенных процедур и лекарств.

Основные бизнес-функции: учет действий персонала больницы по лечению пациентов, расчет стоимости лечения пациента.

Отчеты: стоимость лечения пациентов в больнице за заданный период с возможностью группировки и сортировки по полям, медицинская карта пациента, дальнейшие рекомендации.

Студент может предложить свои темы КР, согласовав их с руководителем.

# РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ. ПОСТАНОВКА ЗАДАЧИ

На первом этапе разработки необходимо описать предметную область, решаемую задачу, все требования, допущения и ограничения. Постановка задачи выполняется на естественном языке. При необходимости для пояснений используются графические изображения. Результатом работ данного этапа является согласованное и утвержденное обеими сторонами техническое задание (ТЗ) на курсовое проектирование. Образец ТЗ приведен в приложении 1.

Целью постановки задачи является согласование с заказчиком (в курсовой работе – это руководитель) выполняемых программой функций, а также показать, что разработчик правильно понял задачу, учитывает все важные для заказчика требования и ограничения.

Для выполнения задания необходимо определить:

* пользователей системы (не менее двух пользователей),
* автоматизируемые задачи (функциональность приложения) для каждого пользователя,
* ограничения целостности данных, накладываемые предметной областью,
* эскизы форм для ввода данных,
* эскизы форм для отчетов.

Данные пункты согласовываются с заказчиком (руководителем) и становятся требованиями к программе после подписания ТЗ.

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – сформулирована тема, содержательно описана предметная область системы; | 2 |
| – указаны пользователи и все функциональные требования (автоматизируемые задачи пользователей) ; | 2 |
| – описан основной бизнес-процесс системы; | 2 |
| – описан интерфейс (пункты меню, кнопки, используемые устройства взаимодействия) ; | 2 |
| – выявлены ограничения целостности данных: уникальные поля, функциональные зависимости и т.п., а также, если нужно, пользовательские ограничения . | 2 |
| *Итого,* максимум | 10 |

ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

Процесс проектирования базы данных разбивается на 3 основных этапа.

* Концептуальное (инфологическое) проектирование – формулирование, анализ и выделение основных требований к данным, как их воспринимает пользователь системы.
* Логическое (даталогическое) проектирование – преобразование концептуальных требований в структуры выбранной модели данных.
* Физическое проектирование – реализация особенностей хранения данных, методов доступа, ограничений средствами используемой СУБД.

## Концептуальная (инфологическая) модель предметной области

Концептуальное проектирование БД решает следующие задачи:

* построение глобальной информационной модели предметной области;
* системный анализ и обсуждение с заказчиком информационной модели;
* проверка непротиворечивости и полноты модели, т.е. ее возможности обеспечивать требуемую функциональность системы.

Первым этапом разработки БД информационной системы является построение модели реальных объектов, явлений или процессов с учетом их взаимосвязей. Здесь под моделью понимается *информационная модель*, т. е. исключительно та информация об объектах, которая хранится в БД и используется ИС для ее функционирования. Концептуальная модель отражает представление пользователя о работе системы, она не зависит от деталей реализации, таких как модель хранения данных, используемая СУБД.

От точности и непротиворечивости концептуальной модели, построенной на данном этапе, зависит качество созданной ИС: ее быстродействие, удобство использования, масштабируемость и т.п. К проектированию ИС на данной стадии следует подойти особенно внимательно, так как допущенные ошибки концептуального проектирования стоят особенно дорого.

Поэтому перед проектированием ИС необходимо четко определить, какие данные использует система, как они связаны между собой, их типы и возможные ограничения, выявить функциональные зависимости.

В сложных системах концептуальные модели могут строиться отдельно для каждой категории пользователей, затем они объединяются в общую глобальную модель. В более простых случаях допускается сразу строить глобальную модель.

Обычно инфологическая модель представляется в виде графической схемы, в нашем случае – виде ER-диаграммы [2]. Мы будем строить ER-диаграмму с применением языка моделирования UML. Скачать редактор UML-диаграмм можно по ссылке <https://www.umlet.com.htm>, существует также online UML-редактор <https://www.umlet.com/umletino/umletino.html>.

Название ER-диаграмма происходит от английского обозначения этого метода «Entity-Relationship», что в переводе означает «*сущность-связь*». Для построения инфологической модели необходимо выделить ***сущности*** предметной области, их ***атрибуты*** и установить ***связи*** между сущностями.

***Сущность*** – некоторый обособленный объект или событие, информацию о котором необходимо сохранять в базе данных, имеющий определенный набор свойств – атрибутов. Сущность характеризуется именем и списком атрибутов. Каждому набору атрибутов соответствует конкретный экземпляр сущности.

Сущность может быть:

* *сильная* – существует сама по себе, например, «Студент»;
* *слабая* – существует в контексте другой сущности, например, «Оценка студента» зависит от сущности «Студент».

Сущность на ER-диаграмме отображается в виде прямоугольника (см. рисунок 1). В верхней отчерченной части указывается имя сущности, ниже помещаются атрибуты сущности.

Имя сильной сущности указывается обычным текстом; слабой – курсивом.

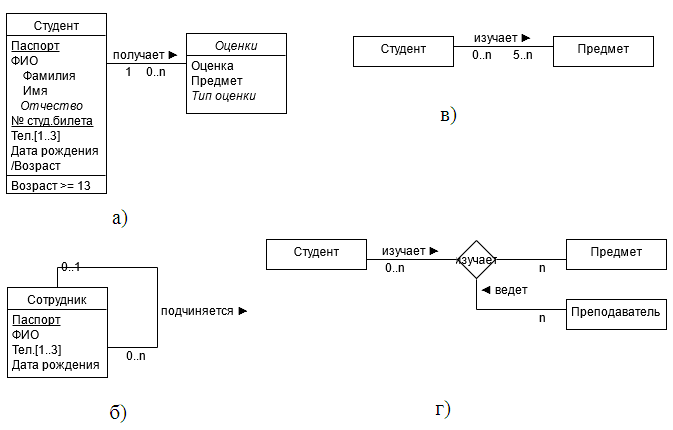


Рисунок 1 – Отображение атрибутов и связей на ER-диаграмме

(а – бинарная связь *один-ко-многим*; б – рекурсивная необязательная связь *один-ко-многим*; в – бинарная связь *многие-ко-многим;* г – *многосторонняя* связь «изучает»)

***Атрибут*** – объект, характеризующий некоторое свойство сущности. Атрибут имеет: имя, тип данных, ограничения, обязательность и уникальность. Пример изображения атрибутов показан на рисунке 1а.

Атрибуты могут быть:

* *простыми* – атрибут имеет единственное значение для данного экземпляра, например, Паспорт, Дата рождения;
* *многозначными* – экземпляр сущности может содержать несколько однотипных атрибутов, например, № телефона;
* *составными* – атрибут представляет собой список из нескольких атрибутов, например, адрес состоит из номера почтового отделения, названия улицы, номера дома и квартиры.

Атрибут также может быть *уникальным* – его значение различно у всех экземпляров сущности, например, серия и номер паспорта. Также атрибут может быть *обязательным* или *не обязательным*, например, номер квартиры в частном доме может отсутствовать.

Особо следует обратить внимание на *вычислимые* атрибуты. Они в базе банных не хранятся, так как их значение может быть вычислено по значениям других атрибутов, но в модели присутствуют, например, возраст вычисляется по дате рождения.

На ER-диаграмме имена атрибутов выглядят в зависимости от типа (таблица 2).

Таблица 2. Нотация атрибутов

|  |  |  |
| --- | --- | --- |
| Тип атрибута | Имя на ER-диаграмме | Пример |
| уникальный | подчеркнутый шрифт | Паспорт |
| простой | обычный шрифт | ФИО |
| многозначный | после имени в квадратных скобках указывается количество значений | Тел.[1..3]  Дети [0..n] |
| составной | ниже основного имени, с отступом перечисляются зависимые атрибуты | Адрес  улица  дом  квартира |
| обязательный | обычный шрифт | Фамилия |
| необязательный | курсив | Отчество |
| вычислимый | слэш перед именем | /Возраст |

***Связи*** характеризуют отношение между экземплярами сущностей. Существует несколько типов связей: «*один-к-одному*», «*один-ко-многим*» и «*многие-ко-многим*». На этапе концептуального моделирования допускается использование связи «многие-ко-многим».

Каждая связь характеризуется степенью (*кратностью*). Степень связи определяется количеством экземпляров сущности, которые охвачены данной связью.

Связи бывают *бинарными* (между двумя сущностями), *рекусивными* (между экземплярами одной сущности) и *многосторонними*, охватывающими более 2 сущностей. Бинарные связи на ER-диаграмме отображаются сплошной линией, можно дополнительно указать направление и имя связи. На концах линии указывается кратность связи. Многосторонняя связь отображается в виде ромба, к которому подходят связи от соответствующих сущностей. Аналогично, кратность связи, указывается около сущностей.

Участие сущности в связи бывает *обязательным* (1..n) и *необязательными* (0..n).

Примеры связей показаны на рис. 1. Связь на рисунке 1а) означает, что оценка выставляется одному студенту (1), а студент может иметь несколько оценок или ни одной (0..n). Связь на рисунке 1в) означает, что студент изучает не менее 5 предметов (5..n), и данный предмет могут изучать несколько или ни одного студента(0..n). Наконец связь на рисунке 1г) показывает, что данный предмет могут вести несколько преподавателей (n), преподаватель может вести несколько предметов (n), студент изучает несколько предметов, но предмет данного преподавателя может не изучать (0..n).

В нижней части изображения сущности можно указать пользовательские ограничения (рис.1а).

Обратите внимание, что концептуальная модель не зависит от модели хранения данных, поэтому на ER-диаграмме, в отличие от реляционной модели:

* допускаются связи «многие-ко-многим»;
* не выделяются первичные и внешние ключи;
* допускаются многозначные, составные атрибуты и вычислимые атрибуты;
* желательно показать атрибуты – потенциальные ключи и пользовательские ограничения.

## Системный анализ концептуальной (информационной) модели

На данном этапе мы проверяем непротиворечивость и полноту модели:

* модель не должна содержать *повторяющихся атрибутов*, так как избыточность данных приводит к аномалиям вставки, удаления и обновления. Например, атрибуты «Возраст» и «Дата рождения» на рис.1а не могут одновременно присутствовать в БД даже в разных сущностях. Поэтому атрибут «Возраст» сделан вычислимым;
* необходимо выделить *все функциональные зависимости* между атрибутами предметной области. Функциональная зависимость A→ B означает, что в каждый момент времени каждому значению А соответствует лишь одно значение В. A и B могут быть составными атрибутами, в этом случае атрибуты перечисляются через запятую. Значение атрибута(ов) A должно быть уникальным. Имена уникальных атрибутов подчеркиваются;
* следует убедиться *в полноте и непротиворечивости модели*, для этого полезно на ER-диаграмме выделить таблицы и связи (граф), участвующие в реализации как минимум основных бизнес-процессов и убедиться, что вся необходимая информация однозначно доступна (рисунок 3).

Функциональные зависимости модели «Студент» (рис. 1а):

|  |
| --- |
| * Паспорт→ ФИО, № студ.билета, Тел., Дата рождения |
| * № студ.билета → ФИО, Паспорт, Тел., Дата рождения |

Рассмотрим типичные ошибки («ловушки») ER-моделирования. Они обычно возникают из-за неправильной интерпретации связей (рисунок 2).

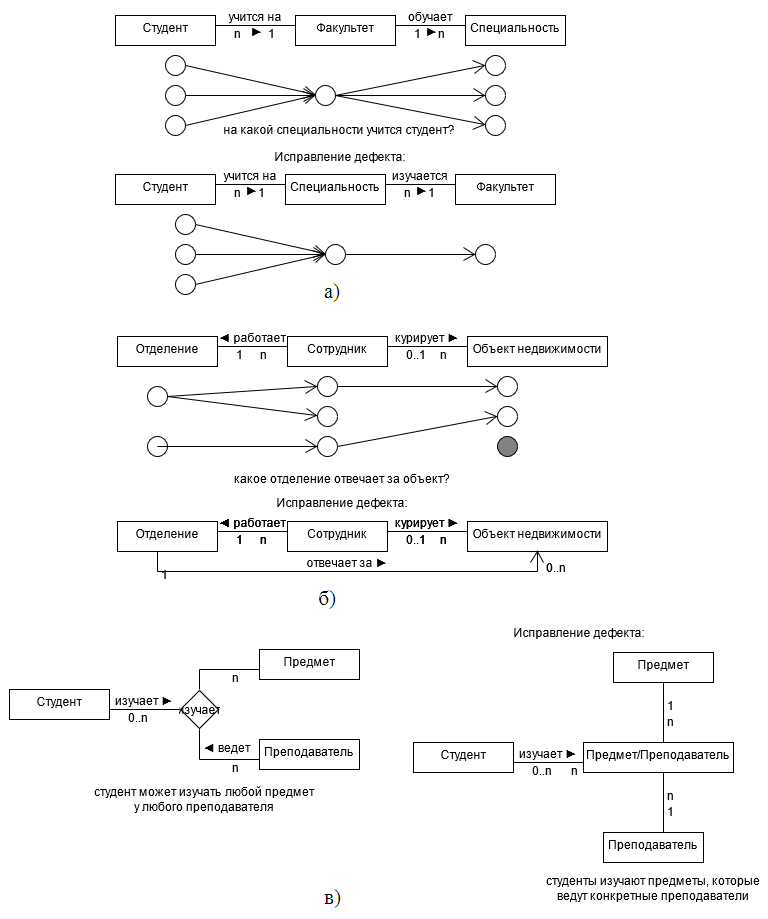


Рисунок 2 – Ловушки ER-моделирования

(а – ловушка разветвления; б – ловушка разрыва; в – ловушка соединения)

***Ловушка разветвления*** имеет место в том случае, если модель отображает связь между сущностями, но путь между отдельными экземплярами этих сущностей однозначно не определяется. Возникает в случае, когда две или больше связей «один-ко-многим» разветвляются из одной сущности. Например, ER-модель на рис. 2а не позволит выяснить «по какой специальности обучается каждый из студентов факультета». Устранить такой дефект можно только путем перестройки исходной модели.

***Ловушка разрыва*** появляется, когда в модели имеются связи между сущностями, но не существует пути между отдельными сущностями. Например, модель на рис. 2б не позволит получить ответ на запрос «какое отделение отвечает за объект недвижимости». Причина дефекта – необязательная связь между Сотрудником и Объектом недвижимости. Для устранения дефекта нужно ввести дополнительную связь между Отделением и Объектом недвижимости.

***Ловушка соединения*** возникает, если необходимо учитывать связи между отдельными экземплярами сущностей. Например, модель на рисунке 2в позволяет связывать любого студента с любым предметом и любым преподавателем. Однако, в действительности, преподаватель читает определенные предметы и именно эти предметы может изучать студент. Устранение дефекта достигается перестройкой модели, как показано на рис. 2в. Вводим сущность-связку Предмет/Преподаватель («преподаватель читает предмет»), а студентов «прикрепляем» к этой связке.

Убедимся, что наша модель обеспечивает требуемую функциональность, проверив выполнение запросов для бизнес-функций.

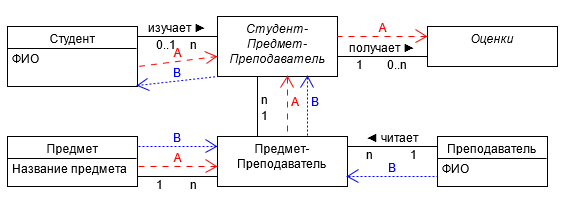


Рисунок 3 – Проверка модели на соответствие функциональным требованиям

(А – «узнать средний балл студента по данному предмету»;  
В – «получить список студентов, изучающих предмет   
у данного преподавателя»)

Последовательность выполнения запроса «узнать средний балл студента по данному предмету» показана стрелками A на рис. 3. Из сущности Студент по имени студента получаем ссылку на экземпляр студента, она единственная. Из сущностей Предмет и Предмет-Преподаватель получаем ссылки на предметы с данным названием, их может быть несколько (n). Сущность Студент-Предмет-Преподаватель фильтруем по ссылкам на студента и предмет, получая ссылки на оценки данного студента по предмету. Осталось выбрать и усреднить оценки из сущности Оценки. Таким образом, запрос выполняется правильно.

Стрелками B на рис. 3 показана последовательность выполнения запроса «получить список студентов, изучающих предмет у данного преподавателя». В данном случае по Названию Предмета и ФИО Преподавателя переходим к связке Предмет-Преподаватель, получим единственный экземпляр сущности Предмет-Преподаватель. Далее по связи переходим к связке Студент-Преподаватель-Предмет и фильтруем ее по предмету и преподавателю. В результате получим список Студентов. Т.е. эта задача тоже выполнима.

Для каждой сущности составляем таблицы, в которых прописываем свойства атрибутов: тип, обязательность, значение по умолчанию, уникальность, пользовательские ограничения. Если пользовательские ограничения затрагивают одновременно несколько сущностей, они прописываются текстом в произвольной форме. Пример оформления отчета по этапу концептуального проектирования приведен в приложении 2.

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – построенаER-диаграмма предметной области; | 5 |
| – приведено описание атрибутов, типов, ограничений; | 4 |
| – выявлены функциональные зависимости; | 3 |
| – выполнена проверка возможности реализации основного бизнес-процесса. | 3 |
| *Итого,* максимум | 15 |

## Построение реляционной модели БД

Цель данного этапа – преобразование концептуальной модели в структуры хранения информации [2]. Результатом этого должна стать логическая, в нашем случае *реляционная модель*, не зависящая от особенностей используемой СУБД. Последовательность действий следующая.

1. **Каждая сущность преобразуется в таблицу** (*отношение*). Каждая таблица должна иметь *первичный ключ* – уникальное поле, значение которого не может быть NULL. Это свойство первичного ключа обеспечивает *целостность сущностей*. В качестве первичного ключа выбираем обязательный уникальный атрибут сущности или создаем суррогатный автоинкрементный первичный ключ (identity). Первичный ключ может быть составным.
2. **Устраняем составные и многозначные атрибуты**. Подчиненные атрибуты можно разместить отдельно в соответствующих полях, либо объединить в одно поле, если бизнес-логика не требует отдельного использования этих атрибутов. На рисунке 4а все поля адреса объединены в одну строку. Возможны два варианта преобразования многозначных атрибутов: а) вынести многозначный атрибут в отдельную таблицу; б) создать несколько полей для нескольких значений атрибута. На рис.4а показан второй вариант. Обратите внимание, что только поле Тел.1 является обязательным.

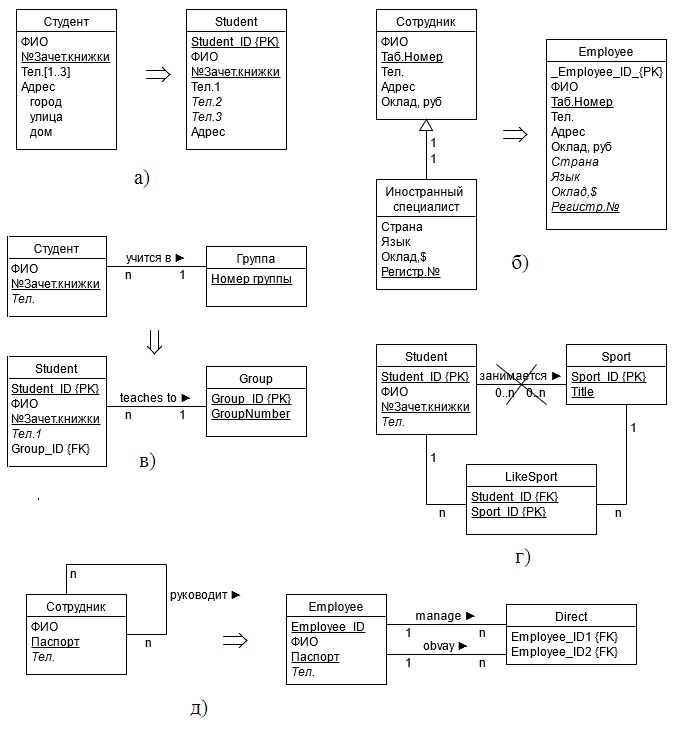


Рисунок 4 – Преобразование концептуальной модели в реляционную

(а – устранение многозначных и составных атрибутов; б – преобразование   
связи 1:1; в – реализация связи 1:n; г – преобразование связи n:n;  
д – преобразование рекурсивной связи n:n)

1. **Создаем связи между таблицами**. Связь реализуется парой ключей: первичный ключ (PK–primary key) в родительской таблице и внешний ключ (FK – foreign key) в дочерней (подчиненной) таблице. Значение внешнего ключа должно указывать на одно из значений первичного ключа или быть NULL. Это свойство внешнего ключа обеспечивает *ссылочную целостность* данных.
2. «*Один-к-одному*» – таблицы можно объединить в одну. Имеет смысл выносить в отдельную таблицу данные, которые необязательны и редко используются (рис.4б). Таблицы связываются по первичным ключам, либо нужно позаботиться о том, чтобы значение внешнего ключа было уникальным.
3. «*Один-ко-многим*» – в дочерней таблице создается поле внешнего ключа {FK} которое ссылается на поле первичного ключа {PK} родительской таблицы. Если на стороне родительской таблицы стоит кратность связи «0..1», то внешнему ключу следует разрешить NULL (рис.4в).
4. «*Многие-ко-многим*» – связь разрывается (рис.4г), создается дополнительная таблица, в которую в качестве внешних ключей помещаются ссылки на первичные ключи связываемых таблиц.
5. . Аналогично поступаем с рекурсивной связью «многие-ко-многим» (рис. 4д).
6. **Реализация пользовательских ограничений**. Современные СУБД предоставляют пользователю средства реализации прочих ограничений, накладываемых предметной областью. Важно, что эти средства работают на стороне сервера. Это – триггеры, и constraints (ограничения). Триггеры и constraints проверяют непротиворечивость данных при вставке, удалении или изменении их в таблице. Перечень необходимых объектов оформляется в виде таблиц, как показано в приложении 2.

## Проверка нормализации БД

Создание любой базы данных преследует две основные цели: понижение избыточности хранимых данных и повышение их надежности [2].

Удовлетворение этих требований достигается нормализацией отношений БД. Процесс нормализации заключается в разложении (декомпозиции) исходных отношений БД на более мелкие и простые. При этом на каждой ступени этого процесса схемы отношений приводятся к нормальным формам (НФ).

В теории реляционных БД разработано несколько нормальных форм, которые подчиняются правилу вложенности.

Отношение находится в первой нормальной форме (1НФ), если значения всех его атрибутов атомарны. Все повторяющиеся группы должны быть удалены в новое отношение.

Отношение находится во второй нормальной форме (2НФ), если оно находится в 1НФ, а каждый его не ключевой атрибут функционально полностью зависит от ключа, т.е. 2НФ требует, чтобы не было не ключевых атрибутов, которые зависят от части первичного ключа.

Отношение находится в третьей нормальной форме (3НФ), если оно находится во 2НФ и все его не ключевые атрибуты зависят только от первичного ключа. 3НФ – это отсутствие транзитивной зависимости, т.е. ни один не ключевой столбец не должен зависеть от другого не ключевого столбца таблицы.

Отношение находится в НФБК, если отсутствуют функциональные зависимости части первичного ключа от не ключевых атрибутов.

При реализации реляционной БД обязательным условием является удовлетворение требований 1НФ. Нормализация до 3НФ сохраняет все функциональные зависимости, т.е. семантику предметной области. НФБК разрушает некоторые функциональные зависимости. Рекомендуется проводить нормализацию до 3НФ. Проверка условий нормализации выполняется отдельно для каждой таблицы по функциональным зависимостям, выделенным в п.4.2.

Пример оформления отчета по данному этапу проектирования приведен в приложении 2.

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – ER-диаграмма преобразована реляционную схему; | 5 |
| – выделены объекты реализации целостности БД; | 3 |
| – проверено соответствие нормальным формам; | 2 |
| *Итого,* максимум | 10 |

## Разработка объектов БД на стороне MS SQL Server

Проектирование физической структуры существенно зависит от выбранной СУБД. Исходными данными для проектирования физической модели являются: *логическая модель, характеристики выбранной СУБД, требования к реактивности системы (время получения пользователем ответа на запрос).*

В этом разделе приводится описание всех объектов БД: таблицы, связи, триггеры, ограничения, хранимые процедуры и пользовательские функции. Создание этих объектов можно выполнить с помощью соответствующих SQL запросов, либо использовать интегрированные среды разработки SQL Management Studio или MS Visual Studio.

Выделим комплекс мер по обеспечению целостности данных, предусмотренный СУБД: обязательные данные, ограничения для доменов полей (целостность приложения), целостность объекта, ссылочная целостность.

Для некоторых столбцов требуется наличие в каждой строке таблицы конкретного и допустимого значения, отличного от *NULL.* Для задания ограничения подобного типа стандарт *SQL* предусматривает использование спецификации *NOT NULL*.

Столбец в таблице имеет собственный домен – набор допустимых значений. Стандарт *SQL* предусматривает различные механизмы определения доменов, например: *CHECK* – позволяет задать перечень требуемых ограничений, *DEFAULT* –позволяет задать значение по умолчанию. В литературе по базам данных механизмы определения доменов часто называют целостностью приложения.

Первичный ключ для каждой строки таблицы должен быть уникальным и иметь непустое значение в каждой строке, что поддерживается с помощью фразы *PRIMARY KEY NOT NULL.* Для потенциальных ключей используется ключевое слово *UNIQUE.* Потенциальный ключ может иметь пустое значение, поэтому фраза *NOT NULL* не обязательна и вставляется по необходимости.

Для обеспечения ссылочной целостности стандарт *SQL* предусматривает механизм определения внешних ключей с помощью предложения *FOREIGN KEY*, а фраза *REFERENCES* определяет имя родительской таблицы. При использовании этого предложения система отклонит выполнение таких *DML-*операций, как *INSERT* или *UPDATE*, с помощью которых будет предпринята попытка создать в дочерней таблице значение внешнего ключа, не соответствующее одному из уже существующих значений первичного или потенциального ключа родительской таблицы. Если пользователь предпринимает попытку удалить или обновить строку в родительской таблице, на которую ссылается одна или более строк дочерней таблицы, язык *SQL* предоставляет возможности, представленные в таблице 3.

Таблица 3 – Спецификация связей

|  |  |
| --- | --- |
| Свойство связи  «Спецификация INSERT,  UPDATE, DELETE» | Действие, выполняемое СУБД при изменении записи в родительской таблице |
| RESTRICT (Нет действия) | Не разрешать удаление (обновление) кортежа в родительской таблице, если имеется хотя бы один кортеж в дочернем отношении, ссылающийся на удаляемый кортеж родительского отношения. |
| CASCADE (Каскадно) | При удалении (обновлении) кортежа в родительской таблице, будут удалены (обновлены) каскадно кортежи в дочернем отношении, ссылающиеся на удаляемый кортеж родительского отношения. |
| SET NULL (Присвоить NULL) | При удалении (обновлении) кортежа в родительской таблице, во всех кортежах в дочернем отношении, ссылающихся на удаляемый кортеж родительского отношения, будет изменено значение внешнего ключа на ***NULL***-значение. |
| SET DEFAULT (Присвоить значение по умолчанию) | При выполнении удаления (обновления) кортежа в родительской таблице, во всех кортежах дочернего отношения, ссылающихся на удаляемый кортеж, изменить значения внешних ключей на некоторое значение, принятое по умолчанию |

Спецификацию CASCADE можно смело ставить на обновление (UPDATE), но каскадное удаление связанных записей может «тихо» удалить большую часть базы. Поэтому следует решить какую спецификацию связи использовать при удалении. Обычно каскадное удаление CASCADE допустимо, если в связи участвует таблица, соответствующая слабой сущности. Во всех остальных случаях ставьте RESTRICT или SET NULL.

Полностью скрипт создания всей физической структуры должен быть приведен в приложении 2 отчета по КР. Проще всего получить его из SQL Management Studio. После создания и заполнения базы данными запустите задачу «Сформировать скрипты…», как показано на рисунке 5.. Затем на странице диалога «Задание параметров скриптов» щелкаете кнопку «Дополнительно» и настраиваете параметры, как показано на рисунке 6. Обратите внимание на выделенные строки, они автоматически не устанавливаются.

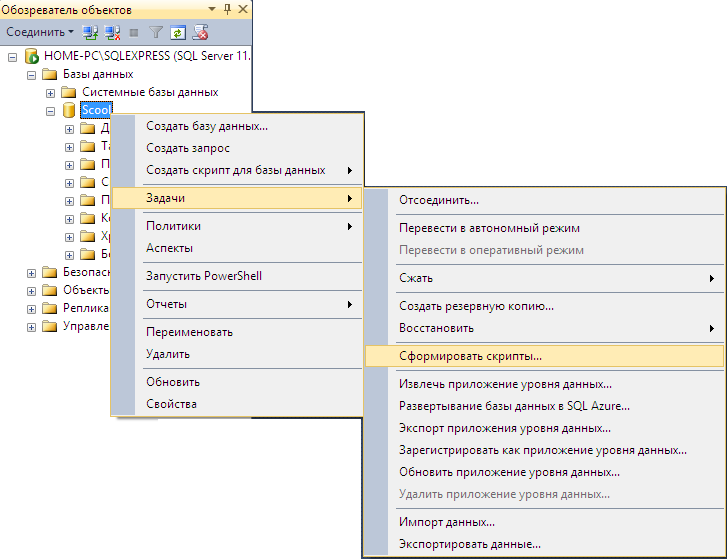


Рисунок 5 – Запуск задачи «Сформировать скрипты…»

Этап физического проектирования требует выделения транзакций – логически законченных последовательностей операций манипулирования данными. Это операции – чтение (*SELECT*), удаление (*DELETE*), изменение (*UPDATE*), добавление (*INSERT*) данных. Каждая из транзакций состоит из отдельных запросов. Необходимо реализовать запросы и получить отчеты, описанные в разделе п.1 «Постановка задачи» отчета по КР. Реализовать транзакции можно в виде хранимых процедур, функций, триггеров.

## Разграничение прав доступа пользователей к БД

Так как база данных является общим ресурсом, к которому обращаются многие приложения и разные категории пользователей, то важной задачей ее проектирования является разработка средств защиты данных от несанкционированного доступа.

*Защищать данные следует средствами СУБД*, т. е. на сервере БД, не рассчитывая на «разумность» программ, взаимодействующих с БД.

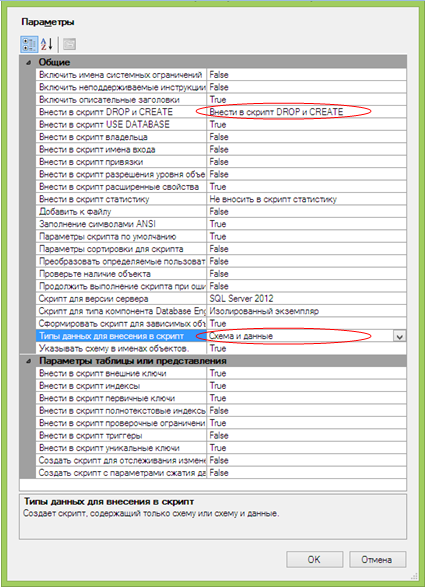


Рисунок 6 – Настройка параметров выгружаемых скриптов

В качестве базовой модели доступа в СУБД Microsoft SQL Server используется двух уровневая ролевая модель. *Субъектами* доступа (участниками) могут быть учетные записи серверного уровня и пользователи баз данных.

1. Учетные записи и *серверные роли* наделяются привилегиями серверного уровня. Каждая из серверных ролей дает возможность выполнять то или иное действие на сервере: устанавливать соединение с сервером, просматривать список имеющихся баз данных, создавать новые базы данных, изменять свойства существующих баз данных или удалять их, управлять настройками сервера (в т.ч. настройками безопасности) и т.п. Чтобы создать серверную учетную запись нужно иметь права администратора ОС или администратора MS SQL Server [5]. Для выполнения курсовой работы на сервере созданы следующие учетные записи с логинами DIRECTOR, OPERATOR, WORKER, INSPECTOR, пароли совпадают с логином, но прописными символами. Эти учетные записи используются для обращения к серверу ***групповых пользователей*** приложения.

Кроме того создана еще одна учетная запись (точка входа на сервер) с именем ГОСТЬ, без пароля. Как показано ниже, эта точка входа может использоваться для авторизации пользователя БД.

1. *Пользователи и роли баз данных* наделяются привилегиями доступа к объектам тех баз данных, в которых они зарегистрированы: привилегиями чтения/записи информации в таблицах, запуска хранимых процедур, изменения свойств объектов базы данных, управления безопасностью на уровне базы данных и т.п. Для их создания достаточно иметь права владельца базы данных. Вы, как создатель базы данных, этими правами обладаете.

4.6.1 Создание пользователей БД и предоставление им доступа

Допустим, что есть пользователь сервера *Mary.* Пока у Mary нет доступа к объектам вашей БД. Последовательность действий:

1. Создание роли БД

CREATE ROLE role\_name;

1. Предоставление прав доступа роли. Инструкции GRANT/REVOKE предоставляют/запрещают доступ к таблицам, представлениям, хранимым процедурам.

USE [TestData];

GO

-- разрешаем SELECT таблицы EmployeeData роли role\_name

GRANT SELECT ON EmployeeData TO role\_name;

GO

-- аналогично для всех объектов

1. Регистрация Mary в качестве пользователя вашей БД testData:

USE [TestData];

GO

CREATE USER [Mary] FOR LOGIN [LOGIN\_Mary];

GO

1. Присоединение Mary к роли

ALTER ROLE role\_name ADD MEMBER [Mary];

GO

1. Создадим хранимую процедуру getPersonalData и дадим Mary право на ее выполнение

USE [TestData];

GO

CREATE PROCEDURE dbo.getPersonalData @User\_ID uniqueidentifier

AS

SET NOCOUNT ON;

SELECT \*

FROM EmployeeData

WHERE User\_ID = @User\_ID

GO

1. разрешаем EXECUTE хранимой процедуры getPersonalData пользователю Mary

GRANT EXECUTE ON getPersonalData TO [Mary];

GO

В данном сценарии Mary имеет доступ только к персональным данным пользователя User\_ID в таблице EmployeeData посредством хранимой процедуры getPersonalData.

Создание ролей и предоставление прав пользователям БД проще выполнить в среде SQL Management Studio [6].

К современным информационным системам часто предъявляется требование разграничения доступа пользователей не только к объектам БД, но и к отдельным записям, с применением авторизации пользователя. Рассмотренный механизм ролей не позволяет сделать это средствами СУБД.

Рассмотрим механизм, позволяющий решить эту задачу средствами СУБД MS SQL Server.

1. Создадим таблицу Users, которая содержит пользователей системы, в том числе их личные User\_Login и User\_Password. Поле первичного ключа User\_ID этой таблицы лучше сделать uniqueidentifier. У каждого пользователя есть также ссылка role\_ID на таблицу Roles – назначенная ему роль в БД.
2. Таблица Roles содержит имена Serv\_Login и пароли Serv\_Password пользователей сервера, сопоставленные с их ролями в приложении. Таблица Roles должна быть недоступна пользователям за исключением администратора БД.
3. Для процедуры авторизации пользователя создается хранимая процедура Check\_Password, принимающая на вход логин и пароль пользователя и возвращающая в случае успешной проверки User\_ID, логин и пароль сервера.

create procedure Check\_PSW @login varchar(50), @pass varchar(50)

AS

BEGIN

SELECT User\_ID, Serv\_Login, Serv\_Pass

FORM Users u INNER JOIN Roles r ON u.Role\_ID=r.Role\_ID

where u.User\_Login=@login and u.User\_Password=@pass;

if @@ROWCOUNT<=0 RAISERROR('Неверен логин или пароль,16,1);

END;

Здесь 16 – степень серьезности ошибки, пользователь могу устанавливать этот уровень от 1-18, эти ошибки по существу являются информационными сообщениями и не влияют на работу сервера. Права на выполнение этой процедуры могут быть назначены любому пользователю. Рекомендуется использовать пользователя ГОСТЬ, так как можно зайти без пароля и этот пользователь не видит ни одной таблицы.

1. Права доступа к таблицам SELECT, INSERT, UPDATE, DELETE и т.п. запрещены всем пользователям БД. Работа с данными осуществляется только через хранимые процедуры. Для всех запросов к данным создаются хранимые процедуры (ХП), предоставляющие своеобразный «интерфейс» для взаимодействия с базой. Процедуры для выборки/изменения персональных данных обязательно требуют идентификатор пользователя User\_ID.

Преимущества и недостатки доступа к БД на основе ролей и хранимых процедур приведены в таблице 4.

Таблица 4 – Свойства методов разграничения прав пользователей

|  |  |
| --- | --- |
| На основе ролей | На основе хранимых процедур |
| Доступ к таблицам целиком | Доступ к отдельным записям в таблицах |
| Пользователь может писать собственные запросы к БД | Пользователь ограничен созданным «интерфейсом» взаимодействия с БД |
| Пользователь должен владеть тонкостями языка SQL и знать структуру БД | Пользователь может не знать структуру БД, знания языка SQL минимальны |
| При изменении схемы БД в приложении придется изменять запросы | При изменении БД администратор подправляет «интерфейс», приложение не затрагивается. |
| Добавление нового функционала не обременительно | Для добавления новых функций требуется обращение к администратору БД |
| Требуется тщательно планировать средства защиты целостности БД: триггеры, ограничения | Часть задач защиты можно возложить на хранимые процедуры |
| При выполнении запросов работает оптимизатор | Оптимизатор ограничен информацией на момент создания ХП |

Как видите, оба метода имеют свои преимущества и недостатки, так что решать вам. Можно использовать гибрид, например, работа с таблицами, содержащими персональные данные, осуществляется с помощью хранимых процедур, в остальных случаях применяются права роли.

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – созданы таблицы, индексы, триггеры, функции в БД; | 6 |
| – настроены связи между таблицами | 4 |
| – созданы роли пользователей БД. | 5 |
| *Итого,* максимум | 15 |

# РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ

## Разработка интерфейса приложения

При разработке пользовательского интерфейса необходимо описать словесно и представить графически алгоритм решения каждой спроектированной задачи, представить и пояснить вид окон и элементов управления (интерфейс) для всех режимов работы (ввод исходных данных, вывод результатов работы).

Проектируемая информационная система должна обеспечить выполнение следующих требований:

* разрабатываемый интерфейс должен включать в себя средства редактирования всех используемых для расчета данных и быть простым и понятным в работе не только для разработчика, но и для обычного пользователя;
* необходимо предусмотреть защиту от некорректного ввода данных во всех формах интерфейса. Если оператор не имеет представления о корректности введенных данных, то в результате возникает множество ошибок, которые приводят к неправильному результату.

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – разработан интерфейс в соответствии с заданием | 6 |
| – создан интерфейс для ввода и вывода всех используемых для расчета данных | 2 |
| – предусмотрена защита от некорректного ввода данных | 2 |
| *Итого,* максимум | 10 |

## Создание и отладка клиентского приложения

Для создания информационной системы используется двух звенная архитектура «клиент-сервер». Клиентское приложение создается в среде MS Visual Studio 2015 и выше, язык программирования С#. В качестве сервера баз данных применяется MS SQL Server 2008 или MS SQL Server 2017, расположенные на сервере кафедры. Для работы с сервером БД приложение использует технологию ADO.NET [3,4].

Обязательной частью приложения является авторизация пользователя с последующим разграничением доступа пользователей к БД. Приложение должно быть отлажено и протестировано, должно обрабатывать ошибки подключения к БД и ошибки пользователя.

***Критерии оценки этапа для БРС:***

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – приложение полностью реализует функционал, указанный в техническом задании | 12 |
| – реализована авторизация пользователей и разграничены права в соответствии с их ролями | 2 |
| – реализована защита от некорректного ввода данных | 4 |
| – реализована обработка ошибок подключения к БД | 2 |
| *Итого,* максимум | 20 |

## Состав программной документации

В приложениях необходимо привести согласованный с преподавателем перечень подлежащих разработке документов в соответствии с требованиями ЕСКД и ЕСПД. Обязательным приложением в отчете является «Исходный текст программы».

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – программная документация содержит необходимые приложения и оформлена в соответствии с требованиями ЕСКД и ЕСПД | 6 |
| – исходный текст программы содержит все модули, документирован | 2 |
| – в документации не выявлено явных противоречий | 2 |
| *Итого,* максимум | 10 |

# СТРУКТУРА И СОДЕРЖАНИЕ ПОЯСНИТЕЛЬНОЙ ЗАПИСКИ

Окончательное содержание отчета определяется самим студентом, но курсовая работа должна содержать обязательные разделы, определяемые преподавателем. Ниже приведены обязательные разделы.

Техническое задание на выполнение курсовой работы (приложение 1)

Пояснительная записка к курсовой работе (приложение 2)

*Титульный лист*

*Аннотация*

*Содержание*

*Введение*

*1. Постановка задачи*

*2. Концептуальное проектирование*

*3. Построение логической модели*

*4. Построение физической модели (описание таблиц, триггеров, ограничений целостности, хранимых процедур и т.п.)*

*5. Руководство пользователя по работе с приложением*

*Заключение (необходимо дать объективную оценку выполненной работе)*

*Библиографический список*

*Приложение 1 Скрипты создания базы данных*

*Приложение 2 Текст программы*

В приложениях обязательными являются текст программы и срипты создания БД. Текст программы должен быть документирован.

В приложения можно включать вспомогательный материал, необходимый для изложения результатов курсовой работы в пояснительной записке, например:

* таблицы вспомогательных цифровых данных;
* иллюстрации вспомогательного характера;
* экраны интерфейса;
* блок-схемы алгоритмов и т.д.

Текст отчета должен быть напечатан на одной стороне стандартного листа белой бумаги формата 210х297 мм (А4) в редакторе W*ord*. Шрифт – *Times New Roman*, размер – 14 pt, интервал одинарный, выравнивание «по ширине». Размеры полей: слева и справа – 2,5 см, сверху – 2 см, снизу – 3 см. Абзацный отступ – 1,25 см.Нумерация страниц – внизу, посередине листа. На титульном листе номер не ставится.

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| – отчет содержит все необходимые разделы | 6 |
| – нет ошибок в оформлении | 2 |
| – нет речевых и грамматических ошибок | 2 |
| *Итого,* максимум | 10 |

# ЗАЩИТА КУРСОВОЙ РАБОТЫ

Защита курсовой работы проходит в открытой форме. На защите могут присутствовать студенты и преподаватели. Оценка по результатам защиты выставляется комиссией из преподавателей, назначаемых заведующим кафедрой.

Для защиты курсовой работы студентом должны быть представлены:

* программа в виде исходного кода и выполняемого файла (архив необходимо заранее загрузить в электронный курс);
* отчет в электронном и печатном виде (файл с отчетом также должен быть загружен в электронный курс);
* презентация доклада (файл с отчетом должен быть загружен в электронный курс).

План презентации:

1 Титульный лист

2 Постановка задачи – 1 слайд

3 Концептуальная модель предметной области– 1-2 слайда

4 Схема базы данных – 1-2 слайда

4 Особенности реализации (за что себя хочется похвалить) – 1-3 слайда

6 Пример использования: скриншоты интерфейса – 1-3 слайда

7 Заключение – 1 слайд

Защита включает в себя доклад студента с демонстрацией презентации перед комиссией в течении 7 минут, и последующие ответы на вопросы (например, «почему было реализовано именно таким образом», «имело ли смысл предусмотреть в программе такие-то функции» и т.п.);

Критерии оценки этапа для БРС:

|  |  |
| --- | --- |
| Содержание работы | Баллы |
| В устном докладе студент показывает знания теории БД, свободно оперирует терминами применительно к рассматриваемой задаче | 6 |
| Подготовлена презентация | 2 |
| Студент может правильно ответить на вопросы членов комиссии | 2 |
| *Итого,* максимум | 10 |

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

* + - 1. Приказ №179 от 25.05.2019г. «Положение о балльно-рейтиноговой системе результатов учебной деятельности обучающегося» [Электронный ресурс] <https://nte.susu.ru/wordpress/wp-content/uploads/2019/06/Polozhenie.pdf> (дата обращения 02.12.2020)
      2. Кузнецов, С. Д. Базы данных: учебник для студ. учреждений высшего проф. образования / С. Д. Кузнецов. – М.: Издательский центр «Академия», 2012. – 496 с.
      3. Евсеева, О.Н. Работа с базами данных на языке C#. Технология АDO.NET: учебное пособие / О. Н. Евсеева, А. Б. Шамшев. – Ульяновск: УлГТУ, 2009. – 170 с. – [Электронный ресурс], URL: [http://venec.ulstu.ru/lib/disk/2009 /Evseeva.pdf](http://venec.ulstu.ru/lib/disk/2009%20/Evseeva.pdf) (дата обращения: 05.12.2020)
      4. Нейгел, К. C# 5.0 и платформа .NET 4.5 для профессионалов / К. Нейгел, Б.Ивьен, Д.Глинн [и др.]. – М.: ООО “И.Д. Вильямс”, 2014. – 1449 с.– [Электронный ресурс], URL: <http://padabum.com/x.php?id=203402>(дата обращения: 05.12.2020)
      5. Настройка разрешений для объектов базы данных [Электронный ресурс], URL: <https://docs.microsoft.com/ru-ru/sql/t-sql/lesson-2-configuring-permis-sions->on-database-objects?view=sql-server-ver15 (дата обращения: 15.12.2020)
      6. Роли [Электронный ресурс], URL: https://professorweb.ru/my/sql-server/2012/level3/3\_12.php (дата обращения: 15.12.2020)
      7. Что такое SQL Server Management Studio (SSMS)? [Электронный ресурс], URL: https://docs.microsoft.com/ru-ru/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver15 (дата обращения: 10.12.2020)

# ПРИЛОЖЕНИЕ 1 Пример технического задания на курсовую работу

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования «Южно-Уральский государственный университет

(национальный исследовательский университет)»  
Институт естественных и точных наук

Факультет математики, механики и компьютерных технологий

Кафедра прикладной математики и программирования

Направление подготовки Прикладная математика и информатика

УТВЕРЖДАЮ

Заведующий кафедрой прикладной математики и программирования

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.А.Замышляева

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2021 г.

ЗАДАНИЕ

на курсовую работу студента

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Иванова Д.С.\_\_\_\_\_\_\_\_\_\_\_\_ Группа \_\_ЕТ-314\_\_\_\_

1. Дисциплина \_\_\_\_\_Администрирование и проектирование хранилищ данных\_

2. Тема работы \_\_\_\_Гостиница\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_Гостиница содержит номера, для которых указывается: класс номера, максимальное число проживающих, стоимость проживания 1 гостя. Ведется учет гостей гостиницы (Ф.И.О., паспорт, дата заезда, предполагаемая дата отъезда, предоплата). Возможно бронирование номеров в гостинице, за которое взимается дополнительная плата от ½ стоимости номера в сутки.

Пользователи системы: менеджер гостиницы, гости

Основные бизнес-функции: учет проживающих в номерах, сдача мест в гостинице гостям.

Отчеты: наличие свободных мест в гостинице и их характеристика, расчеты по оплате за проживание с учетом брони.

3. Срок сдачи студентом законченной работы *25 мая 2021 г.*

4. Перечень вопросов, подлежащих разработке:

* разработка технического задания (цель, инфологическая модель, пользователи БД; автоматизируемые функции)
* построение ER-модели (сущности, связи, атрибуты, функциональные зависимости);
* построение реляционной модели БД; (переход от ER-модели к реляционной, проверка условий нормализации)
* разработка объектов БД на стороне сервера (разграничение прав доступа, домены, таблицы, связи, триггеры, хранимые процедуры);
* разработка интерфейса клиентского приложения для работы с БД
* создание, отладка и тестирование программы;
* оформление программной документации (руководство пользователя, листинг кода) и отчета по курсовой работе.

5. Календарный план

|  |  |  |
| --- | --- | --- |
| Наименование разделов  курсовой работы | Срок выполнения  разделов работы | Отметка руководителя  о выполнении |
| Разработка технического задания | 20.02.2021-04.03.2021 |  |
| Построение ER-модели | 05.03.2021-18.03.2021 |  |
| Построение реляционной модели БД | 19.03.2021-25.03.2021 |  |
| Разработка объектов БД на стороне сервера | 26.03.2021-09.04.2021 |  |
| Разработка интерфейса клиентского приложения для работы с БД | 10.04.2021-15.04.2021 |  |
| Создание, отладка и тестирование программы | 16.04.2021-10.05.2021 |  |
| Оформление программной документации и отчета по курсовой работе | 11.05.2021-24.05.2021 |  |
| Защита курсовой работы | 25.05.2021-30.05.2021 |  |

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Т.Ю.Оленчикова

(подпись)

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Д.С.Иванов\_\_\_\_\_\_\_

(подпись)

# ПРИЛОЖЕНИЕ 2 Пример отчета по курсовой работе

Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Южно-Уральский государственный университет

(национальный исследовательский университет)»

Институт естественных и точных наук

Факультет математики, механики и компьютерных технологий

Кафедра прикладной математики и программирования

|  |  |
| --- | --- |
|  |  |

Успеваемость школьников

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

по дисциплине Администрирование и проектирование   
хранилищ данных

ЮУрГУ–01.03.02.2021.23.ПЗ КР

|  |  |
| --- | --- |
|  | Руководитель работы,  \_\_\_\_\_\_\_\_\_\_/Т.Ю. Оленчикова  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2021 г. |
|  | Автор работы  Студент группы ЕТ-312  \_\_\_\_\_\_\_\_\_\_\_\_\_/ ФИО  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2021 г. |
|  |  |

Челябинск 2021

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

1.1 Инфологическая модель

1.2 Пользователи системы

1.3 Автоматизируемые функции

2 КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

2.1 Построение ER-диаграммы

2.2 Выявление ограничений и правил поддержания целостности данных

2.3 Функциональные зависимости

3 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

3.1 Перевод ER-модели в реляционную форму

3.2 Структура связей БД

3.3 Спецификация ограничений и правил поддержания целостности

3.4 Нормализация БД

4 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

4.1 Права пользователей

4.2 Хранимые процедуры

4.3 Функции

4.4 Триггеры

5 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ

5.1 Подключение к источнику данных

5.2 Рабочая область приложения для преподавателей и руководства

5.3 Рабочая область приложения для учеников

6 ОПИСАНИЕ ПРОГРАММЫ

6.1 Среда разработки

6.2 Минимальные системные требования

6.3 Накладываемые ограничения

ЗАКЛЮЧЕНИЕ

БИБИЛИОГРАФИЧЕСКИЙ СПИСОК

ПРИЛОЖЕНИЕ 1. Исходный текст программы

ПРИЛОЖЕНИЕ 2. Скрипты создания БД

ВВЕДЕНИЕ

Необходимо разработать структуру реляционной базы данных для информационной системы «Успеваемость школьников».

Система предназначена для автоматизации задач, выполняемых учебной частью некоторой общеобразовательной школы. Здесь обеспечивается хранение персональной информации о каждом ученике, данные об учебных дисциплинах, преподавателях, успеваемости школьников по предметам и т.д.

1 ПОСТАНОВКА ЗАДАЧИ

1.1 Инфологическая модель

Стандартная общеобразовательная школа подразделяется на 11 параллелей, каждая из которых может состоять из нескольких классов. В системе содержится информация обо всех классах: номер параллели, буква класса, преподаватели, преподающие у них, список учеников и т.д.

Для каждого сотрудника школы, имеющего доступ к базе данных, хранятся его атрибуты: индивидуальный номер, фамилия, имя, отчество, а также предмет, который он преподает. Любой преподаватель может просматривать отметки ученика любого класса по всем предметам. Однако выставлять оценки он может только по своим предметам и лишь тем классам, в которых он преподает.

Руководству школы (директору, завучам) также предоставлены доступ к информации об успеваемости любого ученика школы и возможность выставлять оценки по своим дисциплинам (если таковые имеются). Кроме этого, они могут формировать отчеты о средних оценках школьников по предметам. Руководство имеет возможность добавлять (удалять) учеников, учителей, классы; вправе изменять информацию о текущей занятости учителей (у каких классов они ведут предмет, и ведут ли они этот предмет вообще).

Сами ученики (либо их родители) могут просматривать только свои оценки.

1.2 Пользователи системы

Учащиеся (родители) (У);

Преподаватели (П);

Руководство (Р).

1.3 Автоматизируемые функции

Просмотр информации об успеваемости всех школьников (П, Р);

Просмотр персональной информации об успеваемости (У);

Выставление (изменение) текущих отметок по предметам (П, Р);

Получение отчетов о средних оценках школьников (Р);

Изменение данных об учениках, классах, преподавателях (Р).

2 КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

Данный раздел посвящен проектированию и нормализации глобальной ER-модели. Строится ее графическое представление, специфицируются ограничения и правила поддержания целостности на уровне глобальной модели.

2.1 Построение ER-диаграммы

Выделено:

* семь сильных сущностей: «Пользователь», «Роль», «Класс», «Учебный план», «Предмет», «Четверть», «Предмет/Учитель»;
* две взаимоисключающие (XOR) дочерние роли «Ученик» и «Учитель», уточняющие сущность «Пользователь»,
* слабая сущность «Оценка» и
* восемь связей типа «один ко многим» (см. рисунок 2.1).

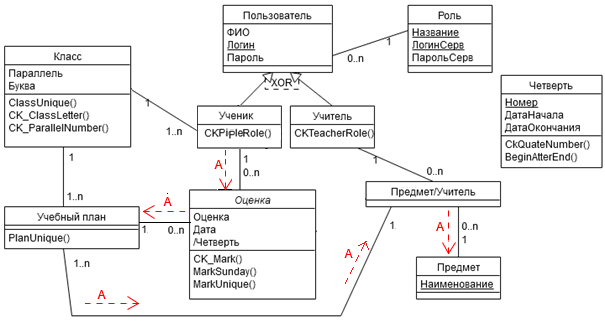


Рисунок 2.1 ­ER-диаграмма

(A–запрос «оценки Ученика по всем предметам»)

2.2 Выявление ограничений и правил поддержания целостности данных

Данный подраздел посвящен выявлению ограничений и правил поддержания целостности базы данных, накладываемых предметной областью.

2.2.1 Сущность «Пользователи»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Логин | Поле должно быть уникально. |

2.2.2 Сущности «Ученик» и «Учитель»

Эти сущности уточняют сущность пользователь. Каждый пользователь может быть либо Учеником, либо Учителем (но не то и другое одновременно). Сущности Ученик соответствует роль «ученик», а сущности Учитель – роль «учитель» или «сотрудник школы»

2.2.3 Сущность «Роль»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Название | Поле должно быть уникально. |
| ЛогинСерв | Поле должно быть уникально в пределах обеих таблиц Ученик и Учитель |

2.2.4 Сущность «Класс»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Параллель | 1 ≤ Параллель ≤ 11. |
| Буква | Поле может принимать значение: «А», «Б», «В», «Г», «Д»…– прописные буквы русского алфавита |

В совокупности атрибуты «Параллель» и «Буква» должны быть уникальны. Таким образом, не могут быть два класса с одинаковыми буквами и параллелями.

2.2.5 Сущность «Учебный план»

Объект содержит ссылки на сущности «Класс» и «Предмет/Учитель». Здесь учитываются следующие ограничения:

* в совокупности все ссылки должны давать уникальную запись.

2.2.6 Сущность «Предмет»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Наименование | Поле должно быть уникально. |

2.2.7 Сущность «Предмет-Учитель»

Объект содержит ссылки на сущности «Учитель» и «Предмет». Здесь учитываются следующие ограничения:

* в объекте содержится ссылка на пользователя, являющегося преподавателем;
* в совокупности все ссылки должны давать уникальную запись.

Сущность Предмет-Учитель является слабой. Ее экземпляр может быть каскадно удален при удалении любой из ссылок Предмет или Учитель

2.2.8 Сущность «Оценка»

Объект содержит ссылки на сущности «Ученик» и «Предмет/Учитель».

|  |  |
| --- | --- |
| Имя  атрибута | Ограничения |
| Оценка | На поле накладывается ограничение 2 ≤ Оценка ≤ 5. Оценка ставится пользователю, который является учеником, и лишь по тому предмету, который ведется в классе этого ученика. |
| Дата | Дата проставления оценки не должна выпадать на воскресенье. |
| /Четверть | Вычислимое поле, определяется по дате выставления оценки и согласовывается с сущностью «Четверть» |

В совокупности все атрибуты должны давать уникальную запись. Т.е. оценка выставляется конкретному ученику, учителем, ведущим данный предмет. Преподаватель не сможет выставить в один день несколько оценок ученику по данному предмету.

Сущность Оценка зависит от сущности Ученик. При удалении Ученика все его оценки также должны быть удалены, однако нельзя удалять сущность Предмет-Преподаватель, если имеются соответствующие Оценки Ученика.

2.2.9 Сущность «Четверть»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Номер | На поле накладывается ограничение 1 ≤ Номер ≤ 4.  Значение поля должно быть уникально.. |
| Дата начала/Дата окончания | На поля накладываются следующие ограничения:  Дата начала не может быть после даты окончания;  Периоды четвертей не могут пересекаться между собой |

2.3 Проверка модели

План выполнения запроса «Показать оценки ученика по всем предметам» показан на рис. 2.1. Проверка этого плана показала, что ER-модель соответствует функциональным требованиям и предметной области.

2.4 Функциональные зависимости

Выявлены следующие функциональные зависимости в предметной области:

|  |  |
| --- | --- |
| Функциональная зависимость | Сущность |
| Параллель, Буква→ | «Класс» |
| Логин→ФИО, Пароль | «Ученик», «Учитель» |
| Название→ЛогинСерв, ПарольСерв  ЛогинСерв→Название, ПарольСерв | «Роль» |
| НаименованиеПредмета→ | «Предмет» |
| ЛогинУченика, ЛогинУчителя, НаименованиеПредмета, Дата→Оценка | «Оценка» |
| Номер→ДатаНачала, ДатаОкончания | «Четверть» |

3 ПОСТРОЕНИЕ ЛОГИЧЕСКОЙ МОДЕЛИ

Данный раздел посвящен проектированию реляционной модели. Здесь выполняется перевод ER-модели в реляционную форму, специфицируются ограничения и правила поддержания целостности на реляционном уровне.

3.1 Перевод ER-модели в реляционную форму

В ER-модели между сущностями нет связей «многие-ко-многим», нет многозначных и составных атрибутов, поэтому дополнительные таблицы не требуются. В реляционной модели каждой сущности соответствует таблица. Было решено для сущностей «Ученик» и «Учитель» использовать отдельные таблицы, при этом логин должен быть уникальным в пределах обеих таблиц.

Обеспечение целостности модели достигается включением триггеров, проверяющих логин пользователя. Эти триггеры, типа AFTER INSERT, AFTER UPDATE, необходимо добавить к таблицам «Ученик» и «Учитель» (проверяем уникальность логина). Кроме того (см. п. 3.1.1) необходим триггеры в таблице Marks («Оценки») для поддержания в непротиворечивом состоянии внешних ключей.

В разработанной реляционной форме глобальной модели используется 8 таблиц. Рассмотрим каждую из них более подробно.

3.1.1 Таблица Puples («Ученики»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | Первичный ключ | Внешний ключ |
| PipleID | uniqueidentifier | NewGuid | + |  | + | + | - |
| Surname | varchar(20) | - | - |  | + | - | - |
| Name | varchar(20) | - | - |  | + | - | - |
| RoleID | int | - | - |  | + | - | + |
| Login | varchar(20) | - | + | >8 симв | + | - | - |
| Password | varchar(20) | - | >8 буквы, цифры | + | - | - |
| ClassID | int | - | - |  | + | - | + |

Для идентификации записей введен суррогатный первичный ключ со спецификацией uniqueidentifier. Тем не менее, уникальность поля Login должна обеспечиваться согласно модели. Внешний ключ ClassID используtтся для связи с таблицей Class.

3.1.2 Таблица Roles («Роль»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | Первичный ключ | Внешний ключ |
| RoleID | int | identity | + |  | + | + | - |
| Name | varchar(20) | - | - |  | + | - | - |
| LoginServ | varchar(20) | - | - | + | + | - | - |
| PasswordServ | varchar(20) | - | - |  | + | - | - |

Данная таблица соответствует ранее описанному объекту «Роль».

Для идентификации записей введен суррогатный первичный ключ со спецификацией identity (автоинкремент). Кандидаты на первичный ключ уникальные поля «Название» и «ЛогинСерв» не используются из-за их символьного формата. За обеспечение уникальности полей «Название» и «ЛогинСерв» отвечают соответствующие уникальные индексы.

3.1.3 Таблица Users («Учитель»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | Первичный ключ | Внешний ключ |
| UserID | uniqueidentifier | NewGuid | + |  | + | + | - |
| Surname | varchar(20) | - | - |  | + | - | - |
| Name | varchar(20) | - | - |  | + | - | - |
| RoleID | int | - | - |  | + | - | + |
| Login | varchar(20) | - | + | >8 симв | + | - | - |
| Password | varchar(20) | - | >8 буквы, цифры | + | - | - |

Для идентификации записей введен суррогатный первичный ключ со спецификацией uniqueidentifier. Тем не менее, уникальность поля Login должна обеспечиваться согласно модели, для этого необходим триггер CHK\_Login. Уникальность поля Login проверяется сразу по двум таблицам Users и Piples.

3.1.4 Таблица Marks («Оценка»)

Данная таблица соответствует ранее описанному объекту «Оценка».

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | Первичный ключ | Внешний ключ |
| MarkID | int | identity | + |  | + | + | - |
| Mark | int | - | - | 2<=Mark<=5 | + | - | - |
| Date | date | getdate() | + |  | - | - | - |
| TSubjectID | int | - |  | + | - | + |
| UserID | int | - |  | + | - | + |

**Примечание**. Остальные таблицы описываются аналогично и здесь не приводятся. Однако в вашем отчете необходимо показать спецификации всех таблиц.

3.2 Структура связей БД

В этом подразделе описана структура связей между таблицами построенной базы данных. Для большей наглядности эти связи представлены с помощью диаграммы БД (см. рисунок 3.1).

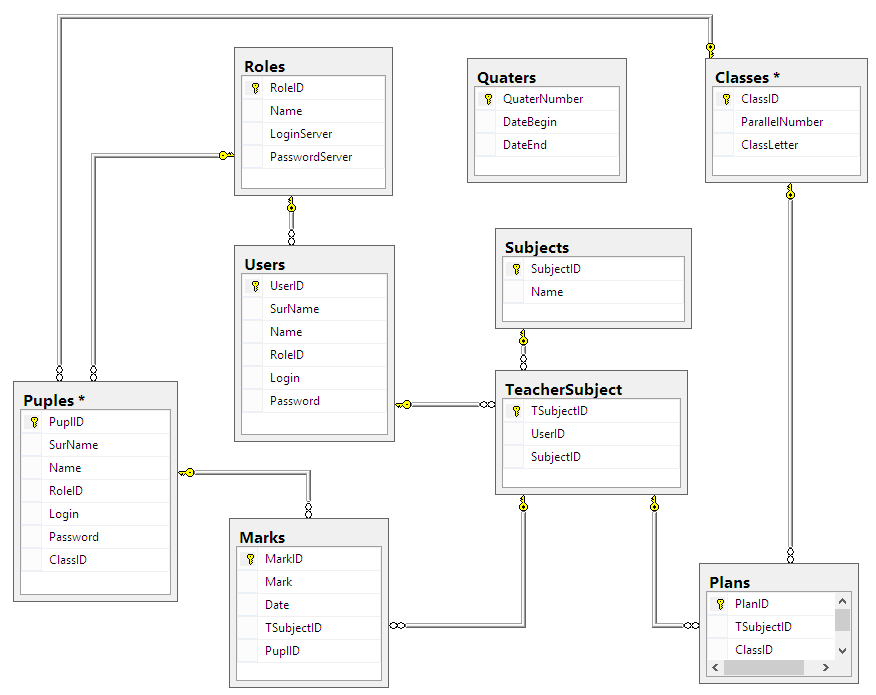


Рисунок 3.1 – Схема реляционной базы данных

3.3 Спецификация ограничений и правил поддержания целостности

В данном подразделе учитываются ограничения и правила поддержания целостности, выявленные на предыдущих этапах в п. 3.1. Новые ограничения и правила обнаружены не были.

3.4 Проверка условий нормализации

Проверка проводится отдельно для каждой таблицы на соответствие нормальным формам.

Все таблицы БД находятся в НФБК тат как:

1НФ – значения всех атрибутов атомарны;

2НФ – отсутствуют функциональные зависимости от части первичного ключа;

3НФ – отсутствуют транзитивные зависимости от первичного ключа

НФБК – отсутствуют функциональные зависимости части первичного ключа от не ключевых атрибутов.

Дальнейшая нормализация БД не требуется.

4 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

В этом разделе подробно описаны используемые хранимые процедуры, табличные и скалярные функции, триггеры, а также права, предоставленные различным группам пользователей.

4.1 Права пользователей

Защита базы данных от несанкционированного доступа осуществляется с помощью создания ролей Ученика, Преподавателя и Руководителя, которые имеют определенные права доступа к таблицам, отдельным столбцам таблиц базы, скалярным и табличным функциям и хранимым процедурам.

Рассмотрим каждую роль и предоставленные ей права подробнее.

4.1.1 Ученик

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Таблица | Права пользователя | | | |
| INSERT | SELECT | UPDATE | DELETE |
| Puples | Запрещено | Разрешено через ХП | Запрещено | Запрещено |
| Users | Запрещено | Разрешено через ХП | Запрещено | Запрещено |
| Subjects | Запрещено | Разрешено | Запрещено | Запрещено |
| Roles | Запрещено | Запрет на отдельные столбцы: LoginServer,  PasswordServer | Запрещено | Запрещено |
| Quarters | Запрещено | Разрешено | Запрещено | Запрещено |
| Plans | Запрещено | Разрешено | Запрещено | Запрещено |
| Marks | Запрещено | Разрешено через ХП | Запрещено | Запрещено |
| Classes | Запрещено | Разрешено | Запрещено | Запрещено |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Права | Хранимые процедуры, функции | | | | | | | | | | | |
| DeleteUser | DeleteSubject | DeletePlan | DeleteClass | MarksTableReport | MarksQuarterForOnePupil | MarksQuarterTable | NameSubjectToID | NameRoleToID | NameClassToID | MarkYear | MarkQuarter |
| EXECUTE | - | - | - | - | - | + | - | + | + | + | - | + |

4.2.2 Преподаватель

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Таблица | | Права пользователя | | | | | | | | | | | | | | |
| INSERT | | | SELECT | | | | | UPDATE | | | | DELETE | | |
| Puples | | Запрещено | | | Запрет на столбцы: Login, Password | | | | | Запрещено | | | | Запрещено | | |
| Users | | Запрещено | | | Разрешено через ХП | | | | | Запрещено | | | | Запрещено | | |
| Subjects | | Запрещено | | | Разрешено | | | | | Запрещено | | | | Запрещено | | |
| Roles | | Запрещено | | | Запрет на отдельные столбцы: LoginServer  PasswordServer | | | | | Запрещено | | | | Запрещено | | |
| Quarters | | Запрещено | | | Разрешено | | | | | Запрещено | | | | Запрещено | | |
| Plans | | Запрещено | | | Разрешено | | | | | Запрещено | | | | Запрещено | | |
| Marks | | Разрешено | | | Разрешено через ХП | | | | | Разрешено | | | | Разрешено | | |
| Classes | | Запрещено | | | Разрешено | | | | | Запрещено | | | | Запрещено | | |
| Права | Хранимые процедуры, функции | | | | | | | | | | | | | | | |
| DeleteUser | | DeleteSubject | DeletePlan | | DeleteClass | MarksTableReport | MarksQuarterForOnePupil | MarksQuarterTable | | NameSubjectToID | NameRoleToID | NameClassToID | | MarkYear | MarkQuarter |
| EXECUTE | - | | - | - | | - | - | - | + | | + | + | + | | - | + |

4.3.3 Руководство

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Таблица | Права пользователя | | | |
| INSERT | SELECT | UPDATE | DELETE |
| Puples | Разрешено | Разрешено через ХП | Разрешено | Разрешено |
| Users | Разрешено | Разрешено через ХП | Разрешено | Разрешено |
| Subjects | Разрешено | Разрешено | Разрешено | Разрешено |
| Roles | Запрещено | Запрет на отдельные столбцы: LoginServer, PasswordServer | Запрещено | Запрещено |
| Quarters | Запрещено | Разрешено | Разрешено | Запрещено |
| Plans | Разрешено | Разрешено | Разрешено | Разрешено |
| Marks | Разрешено | Разрешено | Разрешено | Разрешено |
| Classes | Разрешено | Разрешено | Разрешено | Разрешено |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Права | Хранимые процедуры, функции | | | | | | | | | | | |
| DeleteUser | DeleteSubject | DeletePlan | DeleteClass | MarksTableReport | MarksQuarterForOnePupil | MarksQuarterTable | NameSubjectToID | NameRoleToID | NameClassToID | MarkYear | MarkQuarter |
| EXECUTE | + | + | + | + | + | - | + | + | + | + | + | + |

4.2 Хранимые процедуры

Рассмотрим хранимые процедуры: DeleteUser, DeleteClass, отвечающие за корректное удаление из базы данных пользователей, классов соответственно.

4.2.1 DeleteUser

При удалении пользователя (преподавателя) необходимо учитывать, что также из базы данных должны быть связанные записи в таблице TeacherSubject и в планах установлена в Null ссылка на запись в TeacherSubject (см. листинг 4.1). Параметром процедуры является UserID:

-- Листинг 4.1 Хранимая процедура DeleteUser

CREATE PROCEDURE [dbo].[DeleteUser] @ID uniqueidentifier

AS

BEGIN

UPDATE Plans Set TSubjectID=null

where TSubjectID in

Select TSubjectID from Users inner join TeacherSubject on

Users.UserId=TeacherSubject.User)

where UserID=@ID;

DELETE from TeacherSubject

where [TeacherSubject.UserID=@ID](mailto:TeacherSubject.UserID=@ID);

END

4.2.2 DeleteClass

При удалении класса необходимо учитывать, что также из базы данных должны быть удалены связанные с ним планы, у учеников ссылка на класс должна стать Null. Параметрами процедуры является идентификационный номер класса (см. листинг 4.2).

-- Листинг 4.2 Хранимая процедура DeleteClass

CREATE PROCEDURE [dbo].[DeleteUser] @ID int

AS

BEGIN

DELETE FROM Plan

WHERE ClassID=@ID;

UPDATE Puples SET ClassID=Null

WHERE Puples.ClassID=@ID;

END

4.2.3 CHECK\_PSW

Проверяет логин и пароль пользователя. В случае успеха возвращает логин и пароль сервера для роли и ID пользователя (листинг 4.3). При неудаче генерируется прерывание «Неверен логин или пароль»

-- Листинг 4.3 Хранимая процедура CHECK\_PSW

CREATE PROCEDURE [dbo].[CHECK\_PSW] @user\_ID unigueidentifier,

@login varchar(20), @psw varchar(20)

AS

BEGIN

SELECT u.UserID, r.LoginServer,r.PasswordServer

FROM Roles r INNER JOIN Users u ON r.RoleID=u.RoleID

WHERE u.Login=@login and [u.Password=@psw](mailto:u.Password=@psw)

IF @@rowcout=0 then

SELECT u.PipleID, r.LoginServer,r.PasswordServer

FROM Roles r INNER JOIN Puples u ON r.RoleID=u.RoleID

WHERE u.Login=@login and [u.Password=@psw](mailto:u.Password=@psw);

IF @@rowcout=0 then RAISERROR ('Неверен логин или пароль', 16, 1)

END

4.3 Функции

Реализовано 4 табличных и 5 скалярных функций, основная задача которых – облегчить работу с запросами к БД.

4.3.1 Скалярная функция MarkQuarter

Входные параметры функции – идентификационный номер ученика, предмета и четверти. Возвращает средний балл ученика за четверть по данному предмету (см. листинг 4.3)

-- Листинг 4.3 Средний балл ученика по предмету за четверть

CREATE FUNCTION [dbo].[MarkQuarter]@PupilID uniqueidentifier,

@SubjectID int, @QuarterNumber int

RETURNS numeric(3, 2)

AS

BEGIN

DECLARE @summa float, @i int

SELECT @summa = SUM(Marks.Mark), @i = COUNT(Marks.MarkID)

FROM Marks, Quarters

WHERE Marks.SubjectID = @SubjectID

AND Marks.QuarterNumber = @QuarterNumber

AND Marks.PupilID = @PupilID

AND Quarters.QuarterNumber = @QuarterNumber

AND Marks.Date between Quarters.DateBegin AND

Quarters.DateEnd

RETURN(@summa/@i)

END

4.3.2 Скалярная функция MarkYear

Входные параметры функции – идентификационный номер ученика и предмета. Возвращает средний балл ученика за год по данному предмету (см. листинг 4.4)

-- Листинг 4.4 Средний балл ученика за год по данному предмету

CREATE FUNCTION [dbo].[MarkYear] @PupilID uniqueidentifier,

@SubjectID int)

RETURNS numeric(3, 2)

AS

BEGIN

DECLARE @mark1 float, @mark2 float, @mark3 float,

@mark4 float

SET @mark1 = dbo.MarkQuarter(@PupilID, @SubjectID, 1)

SET @mark2 = dbo.MarkQuarter(@PupilID, @SubjectID, 2)

SET @mark3 = dbo.MarkQuarter(@PupilID, @SubjectID, 3)

SET @mark4 = dbo.MarkQuarter(@PupilID, @SubjectID, 4)

RETURN((@mark1+@mark2+@mark3+@mark4)/4)

END

4.4 Триггеры

Реализованы триггеры: для таблицы «Classes», для таблицы «Marks», для «Plans», для таблицы «Users».

4.4.1 Триггеры таблицы Classes

Триггер ClassUnique не позволяет создавать в таблице классы с одинаковым наименованием (см. листинг 4.8).

--Листинг 4.8 Классы с одинаковым наименованием

CREATE TRIGGER [dbo].[ClassUnique] ON [dbo].[Classes]

FOR INSERT, UPDATE

AS

BEGIN

IF EXISTS(

SELECT Classes.ClassID

FROM Classes, inserted

WHERE Classes.ClassID != inserted.ClassID

AND Classes.ParallelNumber = inserted.ParallelNumber

AND Classes.ClassLetter = inserted.ClassLetter

)

BEGIN

ROLLBACK TRANSACTION

raiserror('Такой класс уже существует',16,1)

END

END

4.4.2 Триггеры таблицы Marks

Триггер MarkUnique не позволяет создавать в таблице несколько оценок со всеми одинаковыми параметрами (см. листинг 4.9).

-- Листинг 4.9 Оценки с одинаковыми параметрами

CREATE TRIGGER [dbo].[MarkUnique] ON [dbo].[Marks]

FOR INSERT, UPDATE

AS

BEGIN

IF EXISTS(

SELECT Marks.MarkID

FROM Marks, inserted

WHERE inserted.MarkID != Marks.MarkID

AND inserted.SubjectID = Marks.SubjectID

AND inserted.PupilsID = Marks.PupilsID

AND inserted.QuarterNumber = Marks.QuarterNumber

AND inserted.Date = Marks.Date

)

BEGIN

ROLLBACK TRANSACTION;

raiserror('Невозможно поставить несколько оценок за

одну дату',16,1);

END

Триггер MarkSunday не позволяет ставить оценки в воскресный день (см. листинг 4.10).

-- Листинг 4.10 Оценки в выходные дни

CREATE TRIGGER [dbo].[MarkSunday] ON [dbo].[Marks]

FOR INSERT, UPDATE

AS

BEGIN

IF EXISTS(

SELECT inserted.MarkID

FROM inserted, Marks

WHERE inserted.MarkID = Marks.MarkID

AND DATEPART(weekday, inserted.Date) = 1

)

BEGIN

ROLLBACK TRANSACTION;

raiserror('Невозможно поставить оценку за эту дату

(воскресенье)', 16, 1);

END

END

4.4.3 Триггеры таблицы Plans

Триггер PlansUnique не позволяет создавать в таблице несколько планов со всеми одинаковыми параметрами (см. листинг 4.11).

-- Листинг 4.11 Уникальность записи Plan

CREATE TRIGGER [dbo].[PlansUnique] ON [dbo].[Plans]

FOR INSERT, UPDATE

AS

BEGIN

IF EXISTS(

SELECT Plans.PlanID

FROM Plans, inserted

WHERE inserted.PlanID != Plans.PlanID

AND inserted.EmployeeID = Plans.EmployeeID

AND inserted.SubjectID = Plans.SubjectID

AND inserted.ClassID = Plans.ClassID

)

BEGIN

ROLLBACK TRANSACTION;

raiserror('Такой план уже существует',16,1);

END

END

Скрипты создания всех хранимых процедур, функций и триггеров приведены в приложении 2 отчета по КР.

5 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ

5.1 Подключение к источнику данных

Работа с программой начинается с подключения к SQL серверу, хранящему спроектированную ранее базу данных. Для подключения к серверу используется форма «Авторизация» (см. рисунок 5.1).

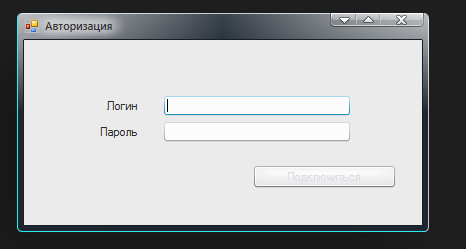


Рисунок 5.1 – Авторизация пользователя

5.2 Рабочая область приложения для учеников

Ученики имеют право на просмотр лишь своей успеваемости. На форме располагается единственная таблица, доступная только для чтения (см. рисунок 5.2).

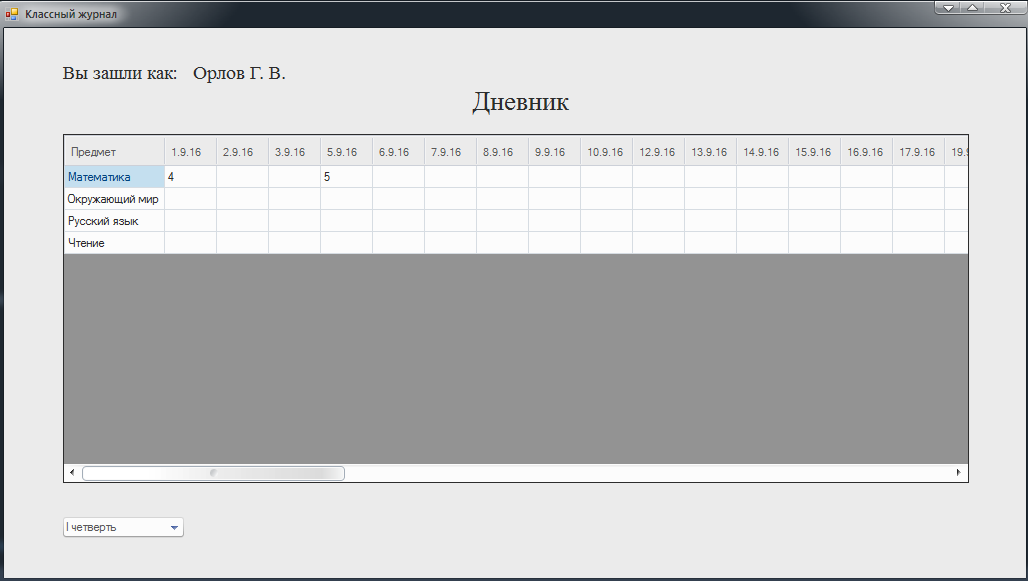


Рисунок 5.2 – Рабочая область для учеников

Здесь должны быть приведены все ваши экранные формы, заполненные данными. Должно быть дано пояснение к формам.

6 ОПИСАНИЕ ПРОГРАММЫ

6.1 Среда разработки

Приложение написано в Visual Studio 2008 на C#. Для создания базы данных использован Microsoft SQL Server 2008.

6.2 Минимальные системные требования

Наличие не менее 256 мегабайт оперативной памяти;

Наличие не менее 50 мегабайт свободного места на жестком диске;

Наличие операционной системы не ниже Windows7;

Наличие установленного .NETFramework не ниже версии 3.5;

Наличие подключения к SQL серверу.

6.3 Накладываемые ограничения

За исключением системных требований никаких других ограничений на программу не накладываются.

ЗАКЛЮЧЕНИЕ

В результате проделанной работы выполнены поставленные задачи, а именно: разработана структура реляционной базы данных для информационной системы «Учет успеваемости школьников», а также спроектировано приложение для работы с этой системой.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Нейгел, К. C# 2008 и платформа .NET 3.5 для профессионалов / К. Нейгел [и др.]– М.: ООО “И.Д. Вильямс”, 2009. – 1392 с.
2. Евсеева, О.Н. Работа с базами данных на языке C#: учебное пособие / О.Н. Евсеева, А.Б. Шамшев. ­– Ульяновск, 2009. – 171 с.

ПРИЛОЖЕНИЕ 1. Код приложения

**П1.1 Конфигурационный файл App.сonfig**

<?xmlversion="1.0" encoding="utf-8" ?>

<configuration>

<configSections>

</configSections>

<connectionStrings>

<add name="WindowsFormsApplication1.Properties. Settings.DataWXConnectionString"

connectionString="Data Source=HOME-PC\МMSSQLSERVER2008;Initial Catalog=DataWX;User ID=ГОСТЬ"

providerName="System.Data.SqlClient" />

</connectionStrings>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework, Version= v4.5" />

</startup>

</configuration>

**П1.2 Форма авторизации**

using System;

using System.Data;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace WindowsFormsApplication1

{

public partial class Form1 : Form

{

public SqlConnection cnn = new SqlConnection();

public Form1()

{ InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

Program.bld.ConnectionString = WindowsFormsApplication1.Properties.Settings.Default.DataWXConnectionString;

textBox1.Text = "IVANOV"; //отладка

}

private void button1\_Click(object sender, EventArgs e)

{

Program.bld.UserID = "ГОСТЬ";

Program.bld.Password = ""; //ГОСТЬ

textBox2.Text = "ivanov"; //отладка

try

{

SqlConnection cnn = new SqlConnection(Program.bld. ConnectionString);

SqlCommand cmd = new SqlCommand("dbo.CHECK\_PSW", cnn);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.Add("@login", SqlDbType.VarChar, 50);

cmd.Parameters["@login"].Value = textBox1.Text;

cmd.Parameters.Add("@psw", SqlDbType.VarChar, 50);

cmd.Parameters["@psw"].Value = textBox2.Text;

cnn.Open();

//выполнить CHECK\_PSW

SqlDataReader dr = cmd.ExecuteReader();

if (dr.Read())

{ // успех - поменяли строку подключения

Program.bld.UserID = dr[4].ToString();

Program.bld.Password = dr["ServPSW"].ToString();

Program.UserCode = (int)dr[0]; Program.FIO = dr[1].ToString();//пользователь

}

else

{ //исключение в случае ошибки

thrownewSystem.Exception("Неверен логин или пароль");

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

return;

}

finally

{ cnn.Close(); //влюбомслучае

}

// сменитьпользователя

if (Program.bld.UserID == "MANAGER")

{

Form2 frm2 = newForm2(this); //окнодляменеджера, this - чтобывернуться

this.Hide();

frm2.Show();

}

else

{

Form3frm3 = newForm3(this); //окно для продавца, this - чтобы вернуться

this.Hide();

frm3.Show();

}

}

private void Form1\_Shown(object sender, EventArgs e)

{ textBox2.Text = "";

}

}

}

Полностью исходный код проекта приведен в электронном виде в папке Н:\Кафедра\Курсовые работы\Год\Администрирование ХД\Фамилия.zip

ПРИЛОЖЕНИЕ 2 Скрипты создания БД

В данном приложении приведены примеры скриптов создания таблиц и связей БД.

**П2.1 Таблица Marks («Оценки»)**

CREATE TABLE [dbo].[Marks](

[MarkID] [int] IDENTITY(1,1) NOT NULL,

[Mark] [int] NOT NULL,

[Date] [date] NOT NULL,

[TSubjectID] [int] NOT NULL,

[PuplID] [int] NOT NULL,

CONSTRAINT [PK\_Marks] PRIMARY KEY CLUSTERED

(

[MarkID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

**П2.2 Таблица Roles («Роли»)**

/\*\*\*\*\*\* Object: Table [dbo].[Poles] Script Date: 09.12.2021 9:08:25 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

SET ANSI\_PADDING ON

GO

CREATE TABLE [dbo].[Roles](

[RoleID] [int] IDENTITY(1,1) NOT NULL,

[Name] [varchar](50) NOT NULL,

[LoginServer] [varchar](50) NOT NULL,

[PasswordServer] [varchar](50) NOT NULL,

CONSTRAINT [PK\_Roles] PRIMARY KEY CLUSTERED

(

[RoleID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

**П2.3 Скалярная функция MarkYear**

CREATE FUNCTION [dbo].[MarkYear](@PupilID int, @SubjectID int)

RETURNS numeric(3, 2)

AS

BEGIN

DECLARE @mark1 float, @mark2 float, @mark3 float, @mark4 float

SET @mark1 = dbo.MarkQuarter(@PupilID, @SubjectID, 1)

SET @mark2 = dbo.MarkQuarter(@PupilID, @SubjectID, 2)

SET @mark3 = dbo.MarkQuarter(@PupilID, @SubjectID, 3)

SET @mark4 = dbo.MarkQuarter(@PupilID, @SubjectID, 4)

RETURN((@mark1+@mark2+@mark3+@mark4)/4)

END

Здесь должна быть приведена распечатка полного скрипта создания БД (см. п.4.5). Вам придется найти и разметить все объекты скрипта, как показано выше.

СОДЕРЖАНИЕ

[1 ЭТАПЫ ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ 3](#_Toc59457178)

[2 ТЕМЫ ДЛЯ КУРСОВЫХ РАБОТ 5](#_Toc59457179)

[3 РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ. ПОСТАНОВКА ЗАДАЧИ 12](#_Toc59457180)

[4 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 13](#_Toc59457181)

[4.1 Концептуальная (инфологическая) модель предметной области 13](#_Toc59457182)

[4.2 Системный анализ концептуальной (информационной) модели 16](#_Toc59457183)

[4.3 Построение реляционной модели БД 19](#_Toc59457184)

[4.4 Проверка нормализации БД 21](#_Toc59457185)

[4.5 Разработка объектов БД на стороне MS SQL Server 22](#_Toc59457186)

[4.6 Разграничение прав доступа пользователей к БД 24](#_Toc59457187)

[5 РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ 29](#_Toc59457188)

[5.1 Разработка интерфейса приложения 29](#_Toc59457189)

[5.2 Создание и отладка клиентского приложения 29](#_Toc59457190)

[5.3 Состав программной документации 30](#_Toc59457191)

[6 СТРУКТУРА И СОДЕРЖАНИЕ ПОЯСНИТЕЛЬНОЙ ЗАПИСКИ 31](#_Toc59457192)

[7 ЗАЩИТА КУРСОВОЙ РАБОТЫ 32](#_Toc59457193)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 33](#_Toc59457194)

[ПРИЛОЖЕНИЕ 1 Пример технического задания на курсовую работу 34](#_Toc59457195)

[ПРИЛОЖЕНИЕ 2 Пример отчета по курсовой работе 36](#_Toc59457196)