Лабораторная работа 1

Основы проектирования структуры БД

1
Выбрать предметную область из вариантов и составить для нее:
а) описание предметной области (от имени конечного пользователя);
б) ER-диаграмму.
Описание и диаграмма включается в отчет по лабораторной работе.

2
Используя MS Access перенести полученную модель в БД, используя таблицы и схему данных.

Прием работы
Прием происходит при наличии оформленного отчета и работающей БД, созданной в среде MS Access.

Вопросы
1. Что такое база данных?
2. Что такое система баз данных?
3. Что такое система управления базами данных?
4. Основное назначение?
5. Основные компоненты СУБД?
6. Что подразумевает понятие абстрагирование в СУБД?
7. Какие существуют уровни абстракции в структурных данных?
8. Опишите уровень представления
9. Опишите концептуальный уровень
10. Опишите физический уровень
11. Виды связей
12. Что такое отношение (таблица) в реляционной модели СУБД?
13. Что такое домен в реляционной модели СУБД?
14. Что такое атрибут (поле) в реляционной модели СУБД?
15. Что такое картеж (хранимая запись) в реляционной модели СУБД?

Варианты заданий

Nпп	Прикладная область	Атрибуты информации
1	Отдел кадров	фамилия сотрудника, имя, отчество, должность, стаж работы, оклад
2	Красная книга	вид живот-го, род, семейство, место обитания, численность популяции
3	Производство	обозначение изд., группа к кот. оно относится, год выпуска, объем выпуска, расход металла
4	Персональные ЭВМ	фирма-изготовитель, тип процессора, тактовая частота, емкость ОЗУ, емкость жесткого диска

5	Библиотека	автор книги, назв., год издания, код УДК, цена, кол-во в библиотеке
6	Спутники планет	Название, название планеты-хозяина, год открытия, диаметр, период обращения
7	Быт студентов	Фамилия студента, имя, отчество, факультет, размер стипендии, число чл. Семьи
8	Спорт. соревнования	Фамилия спортсмена, имя, команда, вид спорта, зачетный результат, штрафные очки
9	С/х работы	фамилия студента, имя, отчество, факультет, вид работ, заработок
10	Сведения о семье	фамилия студента, имя, отчество, факультет, специальность отца, специальность матери, количество братьев и сестер
11	Лесное хозяйство	наименование зеленого массива, площадь, основная порода, средний возраст, плотность деревьев на кв.км
12	Городской транспорт	вид транспорта, номер маршрута, начальная остановка, конечная остановка, время в пути
13	Университет	ФИО и должность преподавателя, назв. предмета, кол-во часов, тип контроля
14	Оптовая база	название товара, количество на складе, стоимость единицы, название поставщика, срок поставки
15	Догов. деятельн. организации	шифр договора, наименование организации, наименование контрагента сроки выполнения, сумма договора, вид договора.
16	Поликлиника	ФИО и дата рождения пациента, ФИО, должность и специализация лечащего врача, диагноз
17	Домоуправление	номер квартиры, общая площадь, полезная площадь, количество комнат, фамилия квартиросъемщика, количество членов семьи, количество детей в семье, есть ли задолженность по квартплате
18	Шахматы	ФИО спортсмена, дата рождения, страна, спортивный разряд, участвовал ли в борьбе за звание чемпиона мира, рейтинг,
19	Ипподром	кличка лошади, масть, возраст, рейтинг, вид забега, фамилия наездника, занятое место
20	Автотранспортное предприятие	номерной знак автом., марка, техн. состояние, грузоподъем-ность, расход топлива, таб. № и ФИО закрепленного водителя

Лабораторная работа 2

Для выполнения этой лабораторной работы необходимо использовать базу данных из лабораторной работы 1.

Значения для запроса вводит пользователь. Во второй части задания количество подсчитывать отдельным запросом, согласно второй части задания.

№ пп	Прикладная область	Атрибуты информации
1	Отдел кадров	фамилия сотрудника, имя, отчество, должность, стаж работы, оклад
	Написать запрос для ФИО сотрудников определенной должности. Должность задается в виде параметра в условии WHERE (например, ‘декан’).	
	Определить количество сотрудников и вывести список, у которых стаж роботы более значения1 лет и оклад меньше значения2.	
2	Красная книга	вид живот-го, род, семейство, место обитания, численность популяции
	Написать запрос для семейства животного определенной среды обитания. Среда обитания задается в виде параметра в условии WHERE (например, ‘пустыня’).	
	Определить количество семейств и вывести список, у которых определенная среда обитания значения1 и количество особей больше значения2.	
3	Производство	обозначение изд., группа к кот. оно относится, год выпуска, объем выпуска, расход металла
	Написать запрос для группы изделий определенного года выпуска. Год выпуска задается в виде параметра в условии WHERE.	
	Определить количество изделий и вывести список, у которых объем выпуска больше значения1 и год выпуска больше значения2.	
4	Персональные ЭВМ	фирма-изготовитель, тип процессора, тактовая частота, емкость ОЗУ, емкость жесткого диска
	Написать запрос для компьютеров с типом процессора определенного года выпуска. Год выпуска задается в виде параметра в условии WHERE.	
	Определить количество компьютеров и вывести список, у которых определенный тип процессора значения1 и год производства больше значения2.	
5	Библиотека	автор книги, назв., год издания, код УДК, цена, кол-во в библиотеке
	Написать запрос для книг определенного года издания. Год издания задается в виде параметра в условии WHERE .	
	Определить количество книг и вывести список, у которых год издания больше значения1 и определенного автора.	
6	Спутники планет	Название, название планеты-хозяина, год открытия, диаметр, период обращения
	Написать запрос для спутников определенной планеты. Планета задается в виде параметра в условии WHERE (например, ‘Юпитер’).	
	Определить количество спутников Юпитера и вывести список, которые открыты в один год значение1 и диаметр которых больше значения2	
7	Быт студентов	Фамилия студента, имя, отчество, факультет, размер стипендии, число чл. Семьи
	Написать запрос для фамилии студента определенного Факультета. Факультет задается в виде параметра в условии WHERE (например, ‘Исторический’).	
	Определить количество студентов и вывести список, у которых количество членов семьи больше значения1 и размер стипендии меньше значения2.	
8	Спорт. соревнования	Фамилия спортсмена, имя, команда, вид спорта, зачетный результат, штрафные очки
	Написать запрос для фамилии спортсмена определенного вида спорта. Вид спорта задается в виде параметра в условии WHERE (например, ‘Легкая атлетика’).	
	Определить количество спортсменов и вывести список, которые состоят в команде значение1 и имеют количество штрафных очков меньше значения2.	

№ пп	Прикладная область	Атрибуты информации
9	С/х работы	фамилия студента, имя, отчество, факультет, вид работ, заработок
	Написать запрос для фамилии студента определенного вида работ. Вид работ задается в виде параметра в условии WHERE (например, ‘администратор’).	
	Определить количество студентов и вывести список, которые задействованы в виде работ значение1 и имеют заработок меньше значения2.	
10	Сведения о семье	фамилия студента, имя, отчество, факультет, специальность отца, специальность матери, количество братьев и сестер
	Написать запрос для фамилии студента у которых отец определенной специальности. Специальность задается в виде параметра в условии WHERE (например, ‘врач’).	
	Определить количество студентов и вывести список, у которых количество братьев и сестер меньше значения1 и специальность матери значения2.	
11	Лесное хозяйство	наименование зеленого массива, площадь, основная порода, средний возраст, плотность деревьев на кв.км
	Написать запрос для основной породы деревьев определенной плотности деревьев на кв. км.. Плотность деревьев задается в виде параметра в условии WHERE .	
	Определить количество зеленых массивов и вывести список, у которых средний возраст больше значения1 и имеют плотность меньше значения2.	
12	Городской транспорт	вид транспорта, номер маршрута, начальная остановка, конечная остановка, время в пути
	Написать запрос для номера маршрута определенного времени в пути. Время в пути задается в виде параметра в условии WHERE	
	Определить количество маршрутов и вывести список, у которых время в пути больше значение1 и номер маршрута больше значения2.	
13	Университет	ФИО и должность преподавателя, назв. предмета, кол-во часов, тип контроля
	Написать запрос для фамилии преподавателя определенного типа контроля. Тип контроля задается в виде параметра в условии WHERE (например, ‘зачет’).	
	Определить количество предметов и вывести список, у которых вид контроля зачет и имеют количество часов меньше значения1.	
14	Оптовая база	название товара, количество на складе, стоимость единицы, название поставщика, срок поставки
	Написать запрос для названия товара определенного поставщика. Поставщик задается в виде параметра в условии WHERE.	
	Перечислить название товара на складе, стоимость которого больше значения1 и срок поставки меньше значения2.	
15	Догов. деятельн. организации	шифр договора, наименование организации, наименование контрагента сроки выполнения, сумма договора, вид договора.
	Написать запрос для наименование организации определенного вида договора. Вид работ задается в виде параметра в условии WHERE.	
	Определить количество договоров и вывести список, у которых шифр меньше значения1 и сумма договора больше значения2.	
16	Поликлиника	ФИО и дата рождения пациента, ФИО, должность и специализация лечащего врача, диагноз
	Написать запрос для ФИО пациента с определенным диагнозом. Диагноз задается в виде параметра в условии WHERE (например, ‘ОРЗ’).	
	Определить количество пациентов и вывести список, с определённым диагнозом у которых дата рождения больше значения1	

№ пп	Прикладная область	Атрибуты информации
17	Домоуправление	номер квартиры, общая площадь, полезная площадь, количество комнат, фамилия квартиросъемщика, количество членов семьи, количество детей в семье, есть ли задолженность по квартплате
	Написать запрос для фамилии квартиросъемщика определенного количества детей. Количество детей задается в виде параметра в условии WHERE (например, ‘2’).	
	Определить количество квартир и вывести список, у которых общая площадь больше значения1 и в них проживает количество детей меньше значения2.	
18	Шахматы	ФИО спортсмена, дата рождения, страна, спортивный разряд, участвовал ли в борьбе за звание чемпиона мира, рейтинг,
	Написать запрос для фамилии спортсменов определенного спортивного разряда. Спортивный разряд задается в виде параметра в условии WHERE (например, ‘мастер спорта’).	
	Определить количество спортсменов и вывести список, у которых дата рождения больше значения1 и их рейтинг меньше значения2.	
19	Ипподром	кличка лошади, масть, возраст, рейтинг, вид забега, фамилия наездника, занятое место
	Написать запрос для клички лошади определенной фамилии наездника. Фамилия наездника задается в виде параметра в условии WHERE.	
	Определить количество лошадей и вывести список, у которых возраст больше значения1 и их рейтинг меньше значения2.	
20	Автотранспортное предприятие	номерной знак автом., марка, техн. состояние, грузоподъем-ность, расход топлива, таб. № и ФИО закрепленного водителя
	Написать запрос для марки автомобиля определенного технического состояния. Техническое состояне задается в виде параметра в условии WHERE .	
	Определить количество марок автомобиля и вывести список, у которых грузоподъемность больше значения1 и расход топлива меньше значения2.	

Прием работы

Прием происходит при наличии оформленного отчета и работающей БД, созданной в среде MS Access.

Контрольные вопросы

1. Дайте определение проекции.
2. Опишите понятие селекции.
3. Через какую операцию реляционной алгебры можно выразить пересечение?
4. Какие разновидности операции «соединение» вы знаете?
5. Какие типичные запросы реализуется с помощью операции деления? Что такое операция деления?
6. Что такое SQL, назначение языка SQL?
7. Назначение команды SELECT?
8. Как применить агрегатную функцию?
9. Чем отличаются WHERE от HAVING?
10. Чем отличаются использование DISTINCT от группировки?

Лабораторная работа 3

Для выполнения этой лабораторной работы необходимо использовать базу данных из лабораторной работы 1.

При выполнении первой части лабораторной смоделировать исключительную ситуацию для функции iif.

№ пп	Прикладная область	Атрибуты информации
1	Отдел кадров	фамилия сотрудника, имя, отчество, должность, стаж работы, оклад
	Произвести выборку сотрудников из двух полей «должность», «фамилия». Если значение поля «должность» в соответствующей таблице не существует, то выводить строку «должность неизвестна» с помощью функции iif.	
	Определить, средний оклад по каждой должности. Вывести ФИО сотрудников, должность у которых оклады выше среднего по должности, вывести также их оклады и их кол-во по должностям.	
	Определить какое количество сотрудников каждой должности получают оклад значение1 (перекрестный запрос).	
2	Красная книга	вид животного, род, семейство, место обитания, численность популяции
	Произвести выборку животных из двух полей «семейство», «место обитания». Если значение поля «место обитания» в соответствующей таблице не существует, то выводить строку «место обитания неизвестно» с помощью функции iif.	
	Определить, среднее количество популяции по каждому семейству. Вывести род, вид животных, у которых популяция ниже среднего по семейству, и их место обитания.	
	Определить по местам обитания, какие виды животных определенного семейства значение1 живут и их популяция больше значения2 (перекрестный запрос).	
3	Производство	обозначение изделий, группа к которым оно относится, год выпуска, объем выпуска, расход металла
	Произвести выборку изделий из двух полей «обозначение изделий», «объем выпуска». Если значение поля «объем выпуска» в соответствующей таблице не существует, то выводить строку «объем выпуска неизвестен» с помощью функции iif.	
	Определить, средний объем выпуска по каждой группе изделий. Вывести обозначение изделий, год выпуска, расход металла, у которых объем выпуска ниже среднего по группе изделий, вывести также их объем выпуска.	
	Определить по группам изделий, какое количество наименований выпущено в году значение1 с расходом металла больше значение2 (перекрестный запрос).	
4	Персональные ЭВМ	фирма-изготовитель, тип процессора, тактовая частота, емкость ОЗУ, емкость жесткого диска
	Произвести выборку компьютеров из двух полей «тип процессора», «емкость ОЗУ». Если значение поля «тип процессора» в соответствующей таблице не существует, то выводить строку «процессор неизвестен» с помощью функции iif.	
	Определить, среднюю емкость жесткого диска по каждой фирме изготовителю. Вывести тип процессора, тактовую частоту, емкость ОЗУ и жесткого диска, у которых емкость жесткого диска выше среднего по фирме изготовителю.	
	Определить кол-во компьютеров по фирмам изготовителям имеют типы процессоров значение1 и емкость жесткого диска меньше значения2 (перекрестный запрос).	
5	Библиотека	автор книги, назв., год издания, код УДК, цена, кол-во в библиотеке
	Произвести выборку сотрудников из двух полей «автор книги», «название». Если значение поля «автор книги» в соответствующей таблице не существует, то выводить строку «автор неизвестен» с помощью функции iif.	
	Определить, среднюю стоимость книг по каждому УДК. Вывести название книги, автор, кол-во книг в библиотеке, у которых цена выше средней по УДК.	
	Определить по кодам УДК, какое количество авторов имеют цену книг больше значения1 и количество экземпляров меньше значения2 (перекрестный запрос).	

№ пп	Прикладная область	Атрибуты информации
6	Спутники планет	Название, название планеты-хозяина, год открытия, диаметр, период обращения
	Произвести выборку спутников из двух полей «название», «год открытия». Если значение поля «год открытия» в соответствующей таблице не существует, то выводить строку «год открытия неизвестен» с помощью функции iif.	
	Определить, по каждой планете хозяину средний диаметр его спутников. Вывести название спутников, диаметр, год открытия при условии, что их диаметр больше среднего диаметра спутников планеты хозяина.	
	Определить по году открытия количество спутников, которые имеют период обращения больше значения1 и диаметр меньше значения2 (перекрестный запрос).	
7	Быт студентов	Фамилия студента, имя, отчество, факультет, размер стипендии, число чл. Семьи
	Произвести выборку студентов из двух полей «Фамилия», «число членов семьи». Если значение поля «число членов семьи» в соответствующей таблице не существует, то выводить строку «сирота» с помощью функции iif.	
	Определить среднее количество членов семьи студента по каждому факультету. Вывести факультет, ФИО студента, размер стипендии, количество членов семьи на экран у которых количество членов семьи больше среднего значения по факультету.	
	Определить по факультетам какое количество студентов имеют размер стипендии меньше значения1 и количество членов семьи больше значения2 (перекрестный запрос).	
8	Спорт. Соревнования	Фамилия спортсмена, имя, команда, вид спорта, зачетный результат, штрафные очки
	Произвести выборку спортсменов из двух полей «имя», «штрафные очки». Если значение поля «штрафные очки» в соответствующей таблице не существует, то выводить строку «штрафные очки отсутствуют» с помощью функции iif.	
	Определить среднее количество штрафных очков по каждому виду спорта. Вывести ФИО спортсмена, команду, зачетный результат, количество штрафных очков, у которых количество штрафных очков больше среднего значения по каждому виду спорта.	
	Определить по именам команд, какое количество спортсменов имеют штрафные очки больше значения1 и вид спорта значение2 (перекрестный запрос).	
9	С/х работы	фамилия студента, имя, отчество, факультет, вид работ, заработок
	Произвести выборку студентов из двух полей «Фамилия», «вид работ». Если значение поля «вид работ» в соответствующей таблице не существует, то выводить строку «нет работы» с помощью функции iif.	
	Определить средний заработок студента по каждому виду работ. Вывести факультет, ФИО студента, размер заработка, вид работ на экран, у которых заработок больше среднего значения по каждому виду работ.	
	Определить по факультетам какое количество студентов имеют размер заработка больше значения1 и вид работ значение2 (перекрестный запрос).	
10	Сведения о семье	фамилия студента, имя, отчество, факультет, специальность отца, специальность матери, количество братьев и сестер
	Произвести выборку студентов из двух полей «Фамилия», «специальность отца». Если значение поля «специальность отца» в соответствующей таблице не существует, то выводить строку «отец безработный» с помощью функции iif.	
	Определить среднее кол-во братьев и сестер студента по каждому факультету. Вывести факультет, ФИО студента, специальности отца и матери, кол-во братьев и сестер на экран у которых кол-во братьев и сестер меньше среднего значения по факультету.	
	Определить по факультетам какое количество студентов, у которых количество братьев и сестер больше значения1 и специальность матери значение2 (перекрестный запрос).	

№ пп	Прикладная область	Атрибуты информации
11	Лесное хозяйство	наименование зеленого массива, площадь, основная порода, средний возраст, плотность деревьев на кв. км
	Произвести выборку наименование зеленого массива из двух полей «площадь», «средний возраст». Если значение поля «средний возраст» в соответствующей таблице не существует, то выводить строку «средний возраст неизвестен» с помощью функции iif.	
	Определить среднее значение площади зеленого массива для каждой породы деревьев. Вывести для каждой породы название зеленого массива, площадь, средний возраст при условии, что площадь больше среднего значения площади зеленого массива для каждой породы деревьев	
	Определить по основным породам деревьев, какое кол-во зеленых массивов, у которых плотность деревьев на кв. км меньше значения1 и площадь меньше значения2 (перекрестный запрос).	
12	Городской транспорт	вид транспорта, номер маршрута, начальная остановка, конечная остановка, время в пути
	Произвести выборку видов транспорта из двух полей «номер маршрута», «время в пути». Если значение поля «время в пути» в соответствующей таблице не существует, то выводить строку «время в пути неизвестно» с помощью функции iif.	
	Определить среднее время в пути по каждому виду транспорта. Вывести вид транспорта, номер маршрута, начальная остановка, конечная остановка, время в пути на экран у которых время в пути меньше среднего значения по виду транспорта.	
	Определить по видам транспорта какое количество маршрутов, у которых время в пути больше значения1 и конечная остановка значение2 (перекрестный запрос).	
13	Университет	ФИО и должность преподавателя, назв. предмета, кол-во часов, тип контроля
	Произвести выборку преподавателей из двух полей «ФИО», «тип контроля». Если значение поля «тип контроля» в соответствующей таблице не существует, то выводить строку «тип контроля неизвестен» с помощью функции iif.	
	Определить среднее кол-во братьев и сестер студента по каждому факультету. Вывести факультет, ФИО студента, специальности отца и матери, кол-во братьев и сестер на экран у которых кол-во братьев и сестер меньше среднего значения по факультету.	
	Определить по факультетам какое количество студентов, у которых количество братьев и сестер больше значения1 и специальность матери значение2 (перекрестный запрос).	
14	Оптовая база	название товара, количество на складе, стоимость единицы, название поставщика, срок поставки
	Произвести выборку товара из двух полей «название товара», «количество на складе». Если значение поля «количество на складе» в соответствующей таблице не существует, то выводить строку «данного товара нет» с помощью функции iif.	
	Определить среднюю стоимость товара на складе по каждому поставщику. Вывести название товара, количество на складе, срок поставки на экран, у которых стоимость товара меньше средней стоимости товара по поставщику.	
	Определить по поставщикам какое количество названий товаров, у которых срок поставки больше значения1 и стоимость единицы товара меньше значения2 (перекрестный запрос)	
15	Догов. деятельн. Организации	шифр договора, наименование организации, наименование контрагента сроки выполнения, сумма договора, вид договора.
	Произвести выборку договоров из двух полей «наименование организации», «наименование контрагента». Если значение поля «наименование контрагента» в соответствующей таблице не существует, то выводить строку «контрагента нет» с помощью функции iif.	
	Определить среднюю сумму договоров по организации. Вывести вид договора, сроки выполнения договора, контрагента, сумма договора на экран у которых средняя сумма договора меньше среднего значения по организации.	
	Определить по видам договоров количество контрагентов, у которых сумма договора больше значения1 и срок выполнения меньше значения2 (перекрестный запрос).	

№ пп	Прикладная область	Атрибуты информации
16	Поликлиника	ФИО и дата рождения пациента, ФИО, должность и специализация лечащего врача, диагноз
	Произвести выборку пациентов из двух полей «диагноз», «ФИО врача». Если значение поля «ФИО врача» в соответствующей таблице не существует, то выводить строку «лечащий врач неизвестен» с помощью функции iif.	
	Определить средний год рождения пациента по каждому врачу. Вывести по ФИО врачей, ФИО пациента, дату рождения, диагноз на экран у которых год рождения больше среднего значения по врачу.	
	Определить по врачам какое количество пациентов, у которых Фамилия начинается на букву значение1 и диагноз значение2 (перекрестный запрос).	
17	Домоуправление	номер квартиры, общая площадь, полезная площадь, количество комнат, фамилия квартиросъемщика, количество членов семьи, количество детей в семье, есть ли задолженность по квартплате
	Произвести выборку квартир из двух полей «количество комнат», «количество детей в семьей». Если значение поля «количество детей в семьей» в соответствующей таблице не существует, то выводить строку «детей нет» с помощью функции iif.	
	Определить среднее кол-во количество членов семьи по кол-ву комнат в квартире. Вывести номер квартиры, общая площадь, фамилия квартиросъемщика, есть ли задолженность на экран у которых кол-во членов семьи меньше среднего кол-во количество членов семьи по кол-ву комнат в квар-ре.	
	Определить по полезной площади квартиры какое кол-во квартир, у которых «задолженность по квартплате» значение1 и кол-во членов семьи больше значения2 (перекрестный запрос).	
18	Шахматы	ФИО спортсмена, дата рождения, страна, спортивный разряд, участвовал ли в борьбе за звание чемпиона мира, рейтинг,
	Произвести выборку спортсменов из двух полей «имя», «спортивный разряд». Если значение поля «спортивный разряд» в соответствующей таблице не существует, то выводить строку «спортивный разряд отсутствует» с помощью функции iif.	
	Определить средний год рождения по каждой стране. Вывести ФИО спортсмена, дата рождения, спортивный разряд, рейтинг, у которых год рождения больше среднего года рождения по стране.	
	Определить по странам, какое количество спортсменов имеют спортивный разряд значение1 и рейтинг меньше значения2 (перекрестный запрос).	
19	Ипподром	кличка лошади, масть, возраст, рейтинг, вид забега, фамилия наездника, занятое место
	Произвести выборку лошадей из двух полей «кличка лошади», «вид забега». Если значение поля «вид забега» в соответствующей таблице не существует, то выводить строку «вид забега неизвестен» с помощью функции iif.	
	Определить средний возраст лошадей по каждому виду забега. Вывести кличка лошади, масть, возраст, вид забега, занятое место на экран у которых средний возраст меньше среднего значения по забегу.	
	Определить по занятым местам какое кол-во лошадей, у которых рейтинг меньше значения1 и масть лошади значение2 (перекрестный запрос).	
20	Автотранспортное предприятие	номерной знак автом., марка, техн. состояние, грузоподъемность, расход топлива, таб. № и ФИО закрепленного водителя
	Произвести выборку автомобилей из двух полей «марка», «ФИО водителя». Если значение поля «ФИО водителя» в соответствующей таблице не существует, то выводить строку «Водитель не прикреплен» с помощью функции iif.	
	Определить среднюю грузоподъемность по каждой марке автомобиля. Вывести номерной знак автом., марка, техн. состояние, грузоподъемность, расход топлива на экран у которых грузоподъемность больше средней грузоподъемности по марке автомобиля.	
	Определить по расходу топлива какое количество автомобилей, у которых техническое состояние значение1 и грузоподъемность меньше значения2 (перекрестный запрос).	

Прием работы

Прием происходит при наличии оформленного отчета и работающей БД, созданной в среде MS Access.

Вопросы

1. Что такое внешнее и внутреннее объединение, чем отличаются?
2. Что такое левое, правое и полное объединение?
3. Что такое перекрестный запрос?
4. Для чего в стандарт SQL2 были введены объединения?

Лабораторная работа №4

Тема: Изменение данных и структуры БД. Клиентский интерфейс для БД.

Цель: Развитие навыков программирования приложений, использующих БД, знакомство с частями SDL и DML языка SQL.

Навыки и умения: модификация данных и определение структуры БД с помощью SQL, использование инструментария MS Access (редактор макросов, VBA модули, конструктор форм), написание клиентского интерфейса.

Теоретическая часть

Структура языка SQL

Все операторы языка SQL можно условно разделить на три группы операторов. Оператор языка запросов – SELECT (был рассмотрен в предыдущих лабораторных работах), операторы языка манипуляции данными (Insert, Update, Delete) и операторы языка определения данных (Create, Drop, Alter).

Запросы DML (ЯМД)

К запросам языка манипуляции данными (Data Manipulation Language) относятся запросы на добавление, удаление и модификацию кортежей.

Добавление кортежа производится командой:

INSERT INTO имя_таблицы [(<список столбцов>)] VALUES (<список значений>)

Список столбцов и список значений указываются через запятую, а значения добавляются в соответствующие столбцы. Если необходимо добавить кортеж целиком (т.е. значения есть для всех полей и их порядок совпадает с порядком полей в отношении), то описание списка столбцов можно опустить.

Пример 1:

Три следующих запроса будут верно исполнены для отношения R1:

INSERT INTO R1(ФИО, Дисциплина, Оценка) VALUES («Попова», «БД», 3);

INSERT INTO R1 VALUES («Попова», «Моделирование», 3);

INSERT INTO R1(ФИО, Дисциплина) VALUES («Бурковский», «Сети ЭВМ»);

Оператор **удаления данных** DELETE позволяет удалить одну или несколько строк из таблицы в соответствии с условиями, которые задаются для удаляемых строк. Синтаксис оператора DELETE следующий:

DELETE FROM <имя_таблицы> [WHERE <условия_отбора>]

Если условия отбора не задаются, то из таблицы удаляются все строки. Операция **обновления данных** UPDATE требуется тогда, когда происходят изменения данных, которые надо отразить в базе данных.

Запрос на обновление может изменить сразу целую группу записей. Этот запрос состоит из трех частей:

- Предложение **UPDATE**, которое указывает на обновляемую таблицу;
- Предложение **SET**, задающее данные для обновления;
- Необязательный критерий **WHERE**, ограничивающий число записей, на которые воздействует запрос на обновление.

Пример 2:

Изменить на 3 оценку по дисциплине «БД» у студента Миронова в таблице R1:

UPDATE R1 SET R1.Оценка = 3
WHERE R1.ФИО = «Миронов» AND R1.Дисциплина = «БД»;

Запросы SDL (ЯОД)

Команды языка определения схемы данных (Schema Definition Language – SDL) представляют собой инструкции SQL, которые позволяют создавать и модифицировать элементы структуры базы данных. Например, используя SDL, можно создавать, удалять таблицы и изменять их структуру, создавать и удалять индексы.

Создание таблицы. Оператор создания таблицы имеет следующий вид:

CREATE TABLE <имя таблицы> (<имя столбца> <тип данных> [NOT NULL] [,<имя столбца> <тип данных> [NOT NULL]] …)

Обязательными операндами оператора являются имя создаваемой таблицы и имя хотя бы одного столбца (поля) с указанием типа данных, хранимых в этом столбце.

При создании таблицы для отдельных полей могут указываться некоторые дополнительные правила контроля вводимых в них значений. Например, конструкция NOT NULL (не пустое) служит для определения обязательного поля.

В табл. 1 перечислены типы данных, которые можно использовать при создании таблиц, используя Microsoft Jet SDL и предложение CREATE (СУБД Access).

Тип данных	SQL тип
Счетчик	COUNTER
Текстовый	TEXT
Memo	LONGTEXT
Денежный	CURRENCY
Дата/время	DATETIME
Числовой (одинарное с плавающей точкой)	SINGLE
Числовой (двойное с плавающей точкой)	DOUBLE
Числовой (целое)	INTEGER
Числовой (длинное целое)	LONG
Числовой (байт)	BYTE

С помощью конструкции **CONSTRAINT** можно задать первичный ключ таблицы.

Пример 3:

Создание таблицы TABL1:

```
CREATE TABLE TABL1 (
[FIL1] COUNTER,
[FIL2] TEXT (10),
[FIL3] CURRENCY, [FIL4] DATETIME,
[FIL5] BYTE, [FIL6] INTEGER,
[FIL7] SINGLE, [FIL8] LONG,
[FIL9] DOUBLE,
CONSTRAINT PrimaryKey PRIMARY KEY ([FIL1]) );
```

В примере 3 поле FIL1 объявлено ключевым, для данного поля создан индекс с именем PrimaryKey.

Похожим образом задается внешний ключ:

Пример 4:

Создание таблицы TABL2:

```
CREATE TABLE TABL2 (
[FIL1] INTEGER,
[FIL2] TEXT (10) NOT NULL,
[FIL3] CURRENCY, [FIL4] LONGTEXT,
CONSTRAINT PrimaryKey PRIMARY KEY ([FIL1],[FIL2]),
CONSTRAINT ForeignKey FOREIGN KEY ([FIL1])
REFERENCES TABL1 ([FIL1]));
```

В данной таблице поле FIL1 объявлено внешним ключом. Между таблицами TABL1 и TABL2 устанавливается связь «один-ко-многим» по полю FIL1.

Для **удаления таблиц** служит инструкция

DROP TABLE <имя таблицы>

Для **модификации структуры таблицы** (добавление, удаление полей, изменения типов полей) используется оператор ALTER TABLE изменения структуры таблицы имеет следующий вид:

ALTER TABLE <имя таблицы>MODIFY | ADD | DROP <имя поля> [<тип данных>]

Создание индексов. Помимо создания индексов в процессе формирования таблицы (с помощью предложения CONSTRAINT), можно также создавать индексы уже после того, как таблица сформирована:

CREATE [UNIQUE] INDEX <имя индекса>ON <имя таблицы> (<имя столбца> [ASC | DESC] [, <имя столбца> [ASC | DESC]…)

Этот оператор позволяет создать индекс для одного или нескольких столбцов заданной таблицы с целью ускорения выполнения запросных и поисковых операций с таблицей. Для одной таблицы можно создать несколько индексов.

Для **удаления индексов** служит инструкция

DROP INDEX <имя индекса> ON<имя таблицы>

Создание форм и отчетов в среде MS Access

СУБД MS Access предоставляет программисту инструментарий для создания форм и отчетов (для пользователя). Соответствующие объекты можно найти среди объектов БД. Доступно как создание форм (отчетов) по определенным таблицам (запросам), так и самостоятельное создание в режиме дизайнера.

Встроенный язык Visual Basic for Application

СУБД MS Access предоставляет возможность описания процедур на языке высокого уровня Visual Basic for Application (VBA). Этот язык встроен во все программные средства, относящиеся к MS Office, и он позволяет работать с объектами БД через выполнение SQL-запросов. Язык VBA является родственником VB и Basic. Также этот язык является процедурным, поддерживает деление на модули, поддерживает дизайнер форм. Обеспечивает обработку исключительных ситуаций и выполнение транзакций. Процедуры и функции, описанные в области видимости public, могут также быть использованы при построении SQL-запросов, подобно встроенным функциям СУБД.

Создать модули можно на соответствующей странице объектов MS Access.

Задание на лабораторную работу

1. Создать базу данных по предметной области своего варианта, которая должна минимум содержать таблицу, состоящую минимум из 6 полей. Реализовать кодовые поля в основной таблице и справочник(и) для расшифровки этих полей. Для создания таблиц БД **использовать скриптовый файл или макрокоманду**, содержащую набор SQL-команд из части языка SDL;
2. Реализовать процедуры Добавления, Удаления, Поиска и Изменения, с помощью SQL;
3. Организовать оконный интерфейс для функций, созданных на предыдущем этапе (добавления, удаления, поиска и изменения);
4. Поиск должен осуществляться с использованием индексов, т.е. поля, по которым осуществляется поиск, должны быть проиндексированы.
5. Организовать вывод результатов в виде отчетов.

Бонус (+ 25%): Организовать механизм авторизации – вход в БД по паролю для нескольких пользователей (статья справки «Пароли (MDB)»).

Прием работы

Прием происходит при наличии оформленного отчета и работающей БД, созданной в среде MS Access.

Вопросы

1. На какие части можно разделить язык SQL, какие команды им соответствуют?
2. Для чего используются индексы?
3. Как обновить несколько полей для нескольких кортежей таблицы одним запросом?
4. Что определяет ключевое слово Constraint?
5. Что такое VBA?
6. Можно ли выполнить добавление данных без указания названия полей, в которые добавляются значения? (почему нельзя или как можно)
7. Как создать форму в MS ACCESS?
8. Как создать отчет в MS ACCESS?

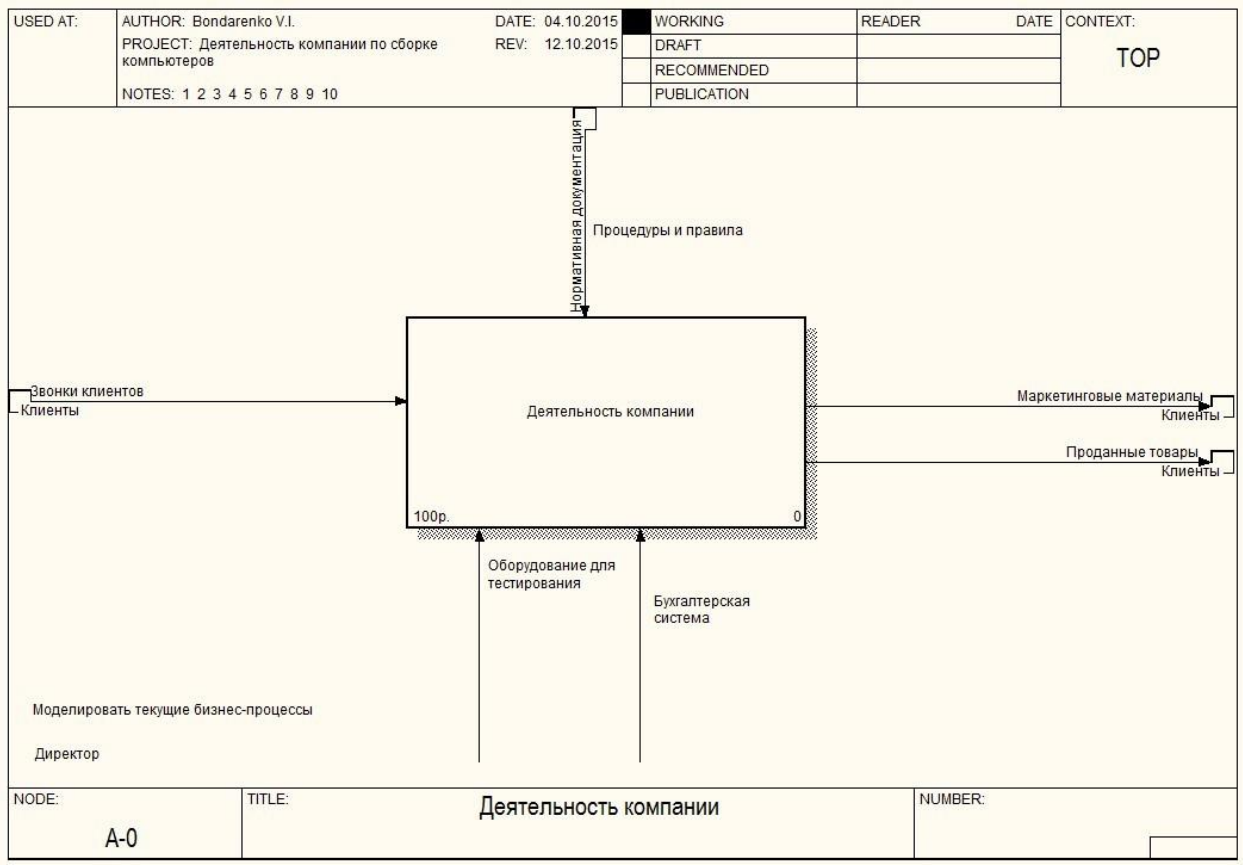
Лабораторная работа № 1

ПОСТРОЕНИЕ МОДЕЛЕЙ ПРЕДМЕТНОЙ ОБЛАСТИ С ИСПОЛЬЗОВАНИЕМ CASE-СРЕДСТВ (*ERWIN PROCESS MODELER, BPWIN ИЛИ RAMUS EDUCATIONAL*)

Цель работы:
Создание функциональной модели системы в нотации IDEF0.

Методика выполнения работы (*на примере Erwin Process Modeler****):***

1. Создадим новую модель на примере деятельности компании по сборке и продаже компьютеров.
2. Разработаем диаграмму верхнего уровня модели (контекстную).



3. Определим функции, на которые может быть разложена функция, обозначенная на контекстной странице модели. Это:
    - Продажи и маркетинг;
    - Сборка и тестирование;
    - Отгрузка и получение.

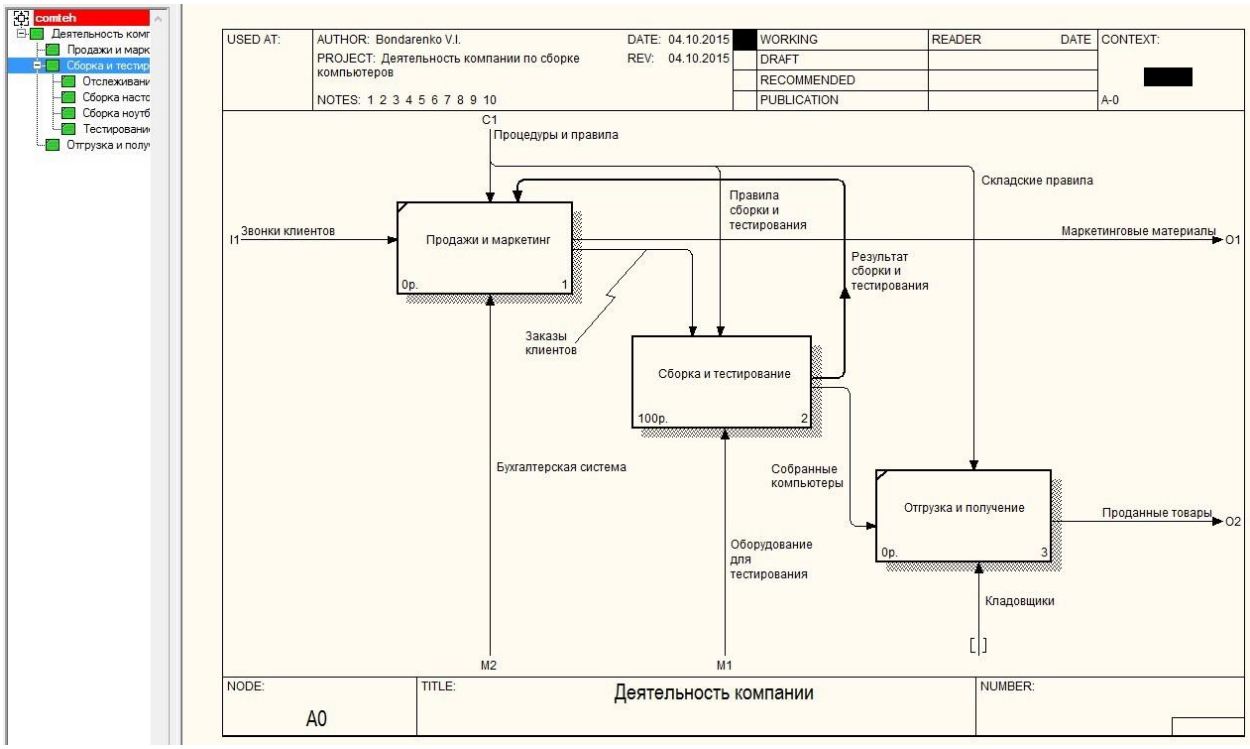
4. Создадим диаграмму декомпозиции первого уровня. Для этого:

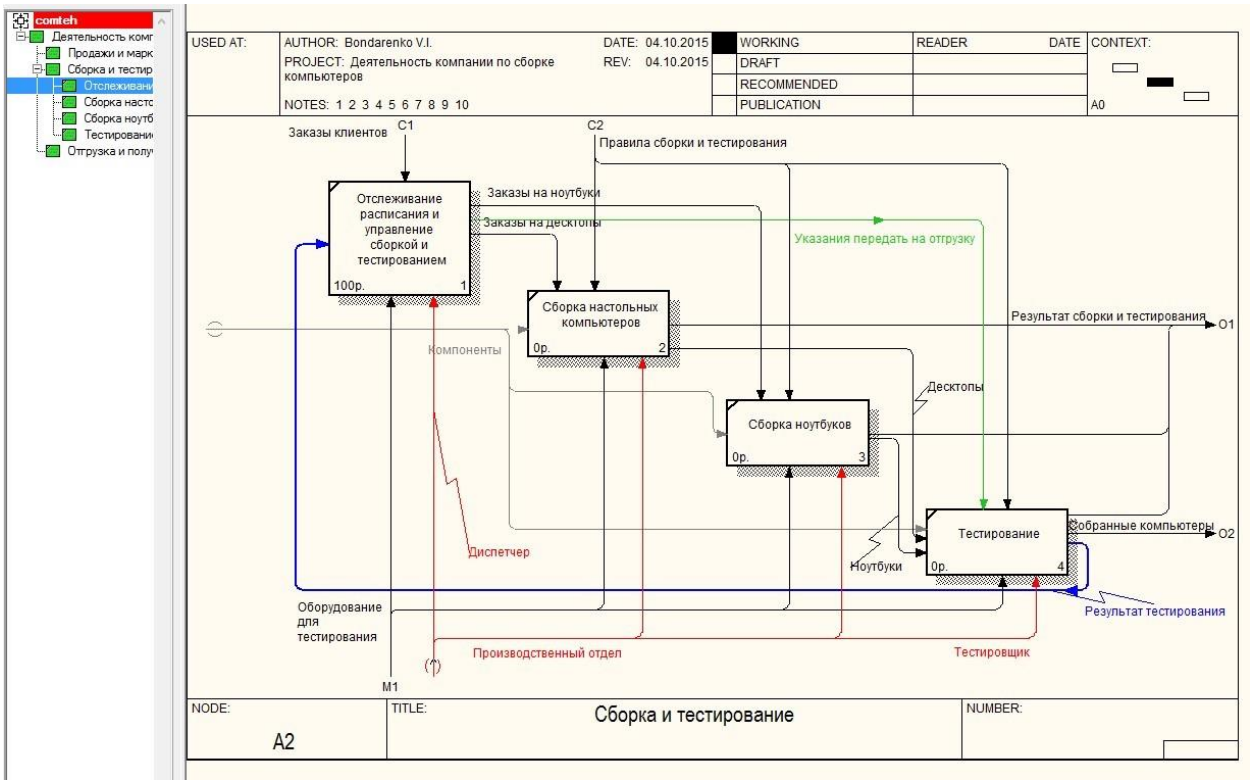
- зададим имя заготовке контекстной диаграммы, выбрав свойства модели (меню Model>Model Properties…), свойства диаграммы – двойной клик мыши на свободном поле диаграммы, или пункт меню Diagram Properties…, или контекстное меню на свободном поле диаграммы.
- зададим свойства модели. На вкладке General зададим информацию о модели. Временные рамки Time Frame примем AS-IS. Это означает, что рассматриваются существующие процессы.
- на вкладке Purpose (Цель) внесем цель моделирования Purpose:

  "Моделировать текущие бизнес–процессы компании" и точку зрения, с которой строится модель Viewpoint: "Директор".
- на вкладке Definition (Определение) задаем определение модели Definition:
  "Учебная модель, описывающая деятельность компании" и границы (рамки) модели Scope: "Общее управление бизнесом компании".
- выделим функциональный блок на контекстной странице;
- на панели инструментов щелкнем по кнопке с изображением черного треугольника, направленного вершиной вниз (декомпозиция)

- в диалоговом окне укажем нотацию создаваемой диаграммы (IDEF0) и число функциональных блоков, которые она должна содержать (3 - по числу выделенных функций).

5. На диаграмме декомпозиции впишем названия выделенных функций в функциональные блоки.
6. Соединим с функциональными блоками интерфейсные дуги, которые мигрировали на созданную диаграмму декомпозиции с контекстной диаграммы.
7. Создадим внутренние дуги для связи функциональных блоков между собой.
8. Аналогично создадим диаграммы декомпозиции для функциональных блоков A1, A2.





Достигнутый результат.

В результате работы средствами редактора Process Modeler создана трехуровневая функциональная модель системы в нотации IDEF0.

Задание

Создайте средствами редактора Modeler трехуровневую функциональная модель в нотации IDEF0 предметной области вашего варианта. Для моделируемой системы в среде Process Modeler (BPwin) должна быть создана трехуровневая функциональная модель, содержащая кроме контекстной диаграммы, диаграммы двух уровней декомпозиции.

1. Создайте новую модель.
2. Разработайте контекстную страницу модели.
3. Обдумайте, на какие функции может быть разложена главная функция системы, обозначенная Вами в функциональном блоке на контекстной странице модели. Помните, что число этих функций должно быть от 3 до 6.
4. Создайте диаграмму декомпозиии первого уровня. При создании диаграммы выберите в диалоговом окне нотацию диаграммы (IDEF0) и укажите, сколько функциональных блоков вы планируете разместить на диаграмме.
5. На диаграмме декомпозиции впишите названия выделенных функций в функциональные блоки. Помните о том, что функциональные блоки на диагонали должны быть расположены в порядке убывания их значимости или в соответствии с последовательностью выполнения работ.
6. Соедините интефейсные дуги, которые мигрировали с диаграммы верхнего уровня на созданную диаграмму декомпозиции в виде стрелок, с функциональными блоками в соответствии с их назначением.
7. Если в этом есть необходимость, сделайте разветвления дуг. Помните о том, что Вы можете оставить единое название для всех веток. В этом случае название располагается до разветвления стрелки. В случае, если ветки обозначают разные объекты, подпишите каждую ветку. 8. Создайте внутренние дуги, связывающие функциональные блоки между собой. Помните, что каждый функциональный блок обязательно должен иметь дуги Управления и Выхода. Дуги Механизма и Входа могут отсутствовать. Именуйте каждую дугу.
9. По описанной выше технологии создайте диаграммы декомпозиции для тех функциональных блоков, прояснить содержание которых требуется по логике модели.

Контрольные вопросы

1. Что такое бизнес-процесс?
2. Перечислите модели структурного подхода к моделированию.
3. Каковы основные компоненты функциональной модели?
4. Что представляют собой методологии функционального моделирования?
5. Укажите преимущества и недостатки объектно-ориентированной методологии моделирования по сравнению с функциональной.

Содержание и оформление отчета

Отчет должен содержать: титульный лист, название и цель работы; вариант задания; скриншоты результатов работы; выводы по работе; ответы на контрольные вопросы.

Лабораторная работа № 2

Проведение системного анализа предметной области.
Разработка логической и физической моделей БД.

Цель работы:

Закрепление и более глубокое усвоение знаний по проектированию информационных моделей.

Методика выполнения работы:

Структурное проектирование с использованием IDEF позволяет построить так называемую модель требований (логическую модель) системы, состоящую из множества взаимоувязанных диаграмм, текстов и словаря данных. Эта модель описывает что должна делать проектируемая система без ссылок на то, как это достигается.

В процессе проектирования системы разрабатываются последовательно ER (сущность-связь), KB (ключевой уровень) и физическая модели.

Эти модели обеспечивают:

- разработку документации базы данных;
- разработку ссылочной целостности БД;
- разработку логической модели БД независимой от конкретного типа СУБД;
- разработку документирования физического проектирования БД в соответствии с бизнес требованиями.

Физическая модель позволяет:

- обеспечить администратору БД достаточность информации, чтобы создать эффективную БД;
- создать контекст для процессов определения и записи в словари данных;
- ассистировать группам приложений в выборе физической структуры программы, которая будет запрашивать данные.

При переходе от логической модели Erwin каждой ее опции ставит соответствующую опцию физической модели.

1 Пример описания проекта.

Задание: разработать информационную модель реализации настольных компьютеров по заказам клиентов.

Для описания объектов предметной области по реализации настольных компьютеров по заказам клиентов выделены следующие сущности: **Тип компьютера, Компьютер, Клиент, Заказ, Продажа, Менеджер, Отдел** (рис. 1)

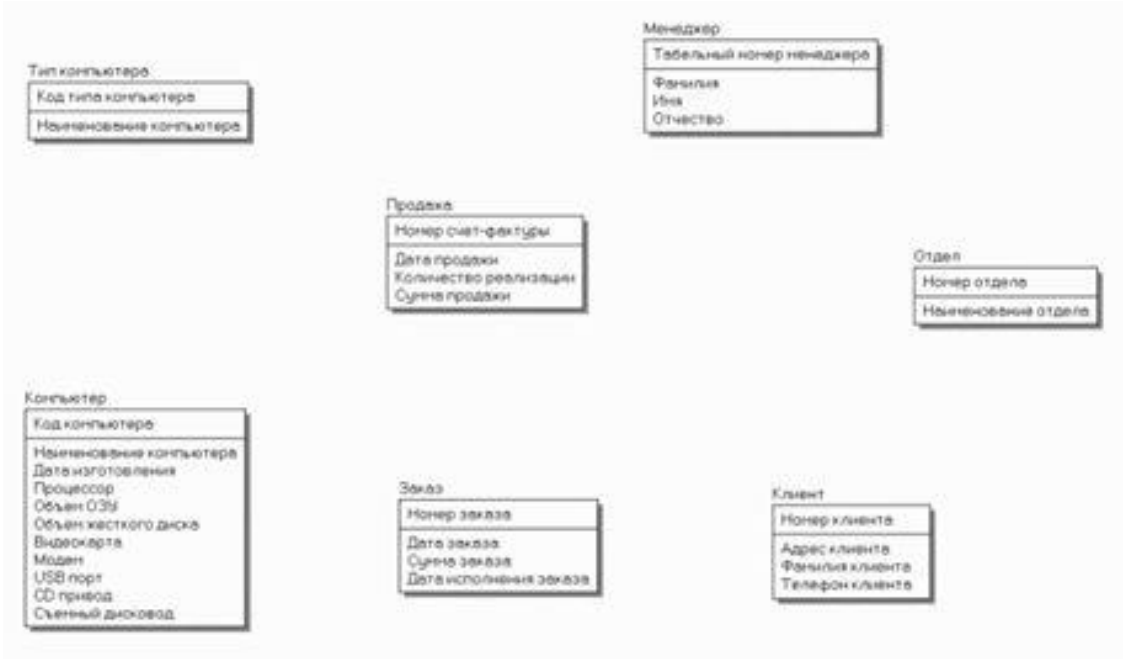


Рисунок 1. Сущности, выделенные для описания объектов предметной области по реализации настольных компьютеров по заказам клиентов

Логическим соотношением между сущностями является *связь*. Каждому виду связи соответствует определенная кнопка, расположенная на палитре инструментов. Имя связи выражает некоторое ограничение или бизнес-правило и облегчает чтение диаграммы. Каждая связь должна именоваться глаголом или глагольной фразой. Например, связь между сущностями **Компьютер**, **Продажа** и **Менеджер** показывает (рис. 2), какой компьютер продан и какой менеджер оформил сделку продажи компьютера: − каждый **Компьютер** < продается > **Продажа**; − каждая **Продажа** < оформляет > **Менеджер**.

Рисунок 2. Связь между сущностями Компьютер, Продажа и Менеджер

В нотации IDEF1X различают *зависимые* и *независимые* сущности. Тип сущности определяется ее связью с другими сущностями. *Идентифицирующая связь* устанавливается между независимой (родительский конец связи) и зависимой (дочерний конец связи) сущностями и показывается на диаграмме сплошной линией с жирной точкой на дочернем конце связи. При установлении идентифицирующей связи ERwin автоматически преобразует дочернюю сущность в зависимую сущность. Зависимая сущность на диаграмме изображается прямоугольником со скругленными углами, например, сущность **Продажа** (см. рис. 2). Экземпляр зависимой сущности определяется только через отношение к родительской сущности. Например, информация о продаже не может быть внесена и не имеет смысла без информации о проданном компьютере. При установлении *идентифицирующей связи* атрибуты первичного ключа родительской сущности автоматически переносятся в состав первичного ключа дочерней сущности.

Операция дополнения атрибутов дочерней сущности при создании связи называется *миграцией атрибутов*. В дочерней сущности новые (мигрированные) атрибуты помечаются как внешний ключ (FK). В случае идентифицирующей связи при генерации схемы базы данных атрибутам внешнего ключа присваивается признак NOT NULL, что означает невозможность внесения записи в таблицу, соответствующей дочерней сущности, без идентификационной информации из таблицы, соответствующей родительской сущности. Например, невозможность внесения записи в таблицу продаж без информации об идентификационном номере проданного компьютера, определенного в таблице описания имеющихся в наличии компьютеров.

Неидентифицирующая связь показывается на диаграмме пунктирной линией с жирной точкой и служит для установления связи между независимыми сущностями. При установлении неидентифицирующей связи дочерняя сущность остается независимой, а атрибуты первичного ключа мигрируют в состав неключевых атрибутов родительской сущности (рис. 3).

Рисунок 3. Пример неидентифицирующей связи между независимыми сущностями **Менеджер** и **Отдел**

Для *создания новой связи* необходимо:

− установить курсор на кнопке, расположенной на палитре инструментов и соответствующей требуемому виду связи, и нажать левую кнопку мыши;

− щелкнуть левой кнопкой мыши сначала по родительской, а затем по дочерней сущности.

Изменить форму линии связи и ее местоположение между связанными сущностями можно путем захвата мышью выделенной линии связи и переноса ее с места на место, пока линия не примет необходимые местоположение и форму.

Редактирование свойств связи осуществляется в диалоговом окне **Relationship**, которое открывается через пункт **Relationship Properties** контекстного меню, активизируемого посредством нажатия правой кнопки мыши на выделенной связи.

Вкладка **General** диалогового окна **Relationship** позволяет задать мощность, имя и тип связи.

Мощность связи (*Cardinality*) служит для обозначения отношения числа экземпляров родительской сущности к числу экземпляров дочерней сущности. Различают 4 типа мощности связи:

− *общий случай* – не помечается каким-либо символом и соответствует ситуации, когда одному экземпляру родительской сущности соответствуют 0,

1 или много экземпляров дочерней сущности;

− *символом P помечается случай*, когда одному экземпляру родительской сущности соответствуют 1 или много экземпляров дочерней сущности, т.е. исключено нулевое значение;

− *символом Z помечается случай*, когда одному экземпляру родительской сущности соответствуют 0 или 1 экземпляр дочерней сущности, т.е. исключены множественные значения;

− *цифрой помечается случай* точного соответствия, когда одному экземпляру родительской сущности соответствует заранее заданное число экземпляров дочерней сущности.

Результат разработки логической модели данных системы "Реализация средств вычислительной техники", предназначенной для учета продаж настольных компьютеров по заказам клиентов приведен на рис. 4.

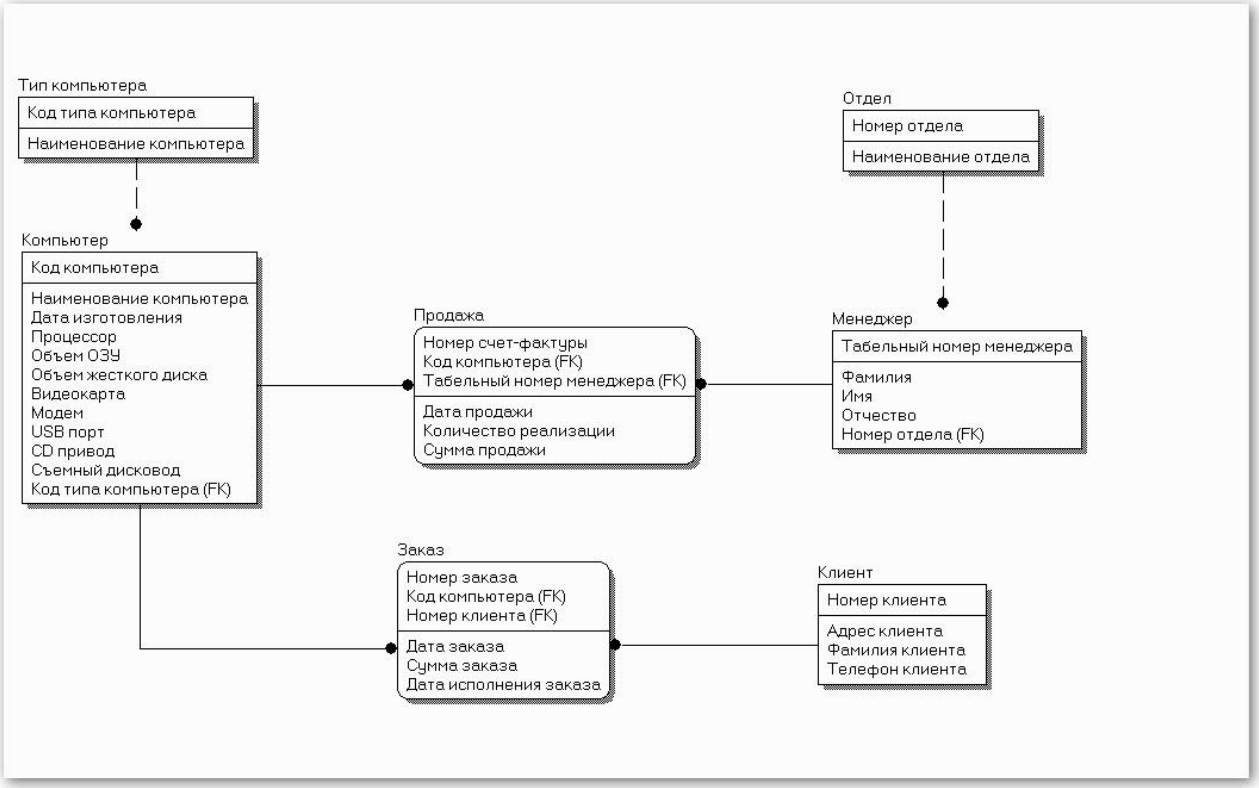


Рисунок 4. Логическая модель данных системы «Реализация средств вычислительной техники»

Созданная логическая модель данных системы является основанием для создания физической модели данных под выбранную СУБД.

Физический уровень представления модели зависит от конкретной реализации СУБД, поэтому необходимо предварительно осуществить ее выбор. CA ERwin Data Modeler 7.3 поддерживает 17 наиболее распространенных СУБД. Выбор СУБД осуществляется в окне **Target Database** диалога **Create Model**. Для выбора СУБД необходимо открыть выпадающий список с перечнем поддерживаемых СУБД и щелкнуть по соответствующему имени. При этом в поле **Version** отобразится версия выбранной СУБД.

В случае автоматического перехода к физической модели ERwin генерирует имена таблиц и колонок по умолчанию на основе имен соответствующих сущностей и атрибутов логической модели, учитывая максимальную длину имени и другие синтаксические ограничения, накладываемые выбранной СУБД. При этом правила ссылочной целостности, принятые на логическом уровне модели данных системы, также принимаются по умолчанию, т.е. сохраняются.

Построение физической модели данных для системы "Реализация средств вычислительной техники" осуществлено путем автоматического перехода от логической модели к физической модели, так как при создании логической модели данных системы был выбран логико – физический тип модели.

Физическая модель данных системы "Реализация средств вычислительной техники" приведена на рис. 5.

Для генерации кода создания базы данных можно использовать пункт главного меню Tools/Forward Engineer.

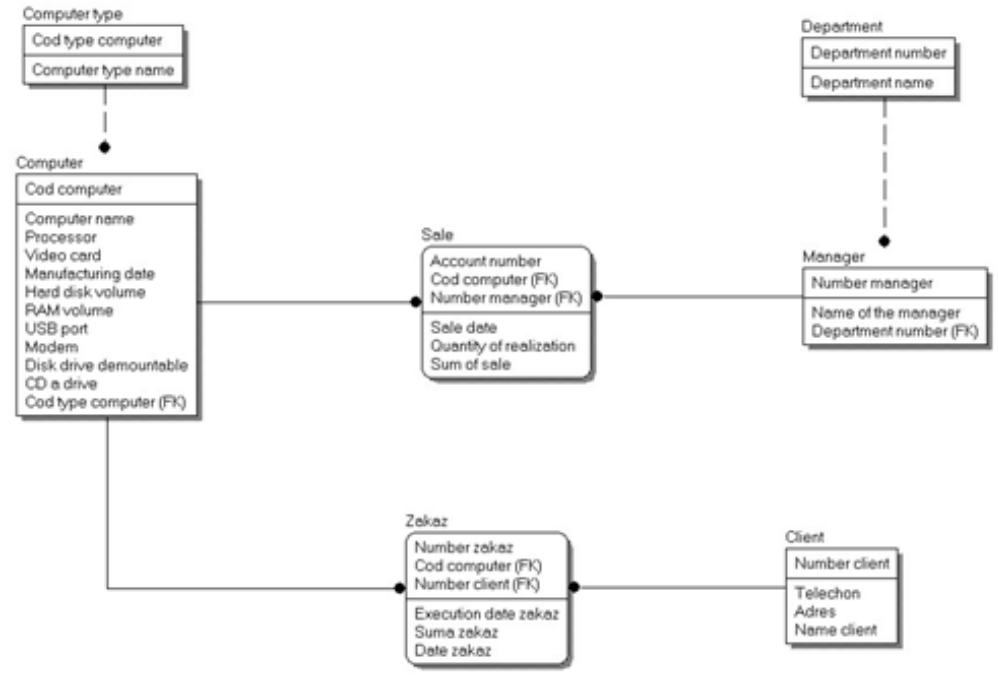


Рисунок 5. Физическая модель данных системы "Реализация средств вычислительной техники"

Порядок выполнения работы.

Настройка работы.

1 Запускается CASE - система Erwin.

2 Осуществляется настройка работы для работы с русским языком и конкретным СУБД.

3 Осуществляется настройка Ссылочной Целостности (Referential Integrity). Для этого из скролинг- меню (Identifying) выбираются опции для отношений IDENTIFYING (Ch.Del- Par.Upd) соответственно (N, Restr., Restr., Restr., None, Restr.). (Здесь опции Cascad указывает, что все атрибуты, указанные в сущности родителя для удаления, одновременно удаляются и у наследуемых сущностей).

Логическое проектирование.

Задача1. Проектирование ER диаграммы.

Цель: построение диаграммы уровня “сущность- связь” и глоссария к ней.

Последовательность действий:

- выделить сущности и присвоить им уникальные имена;
- занести в глоссарий модели формальное определение имен сущностей.

Требования к диаграмме и глоссарию:

- сущности должны быть представлены на диаграмме только именами;
- допускаются зависимые и независимые сущности;
- глоссарий должен содержать краткие формальные определения сущностей.

Порядок разработки ER диаграммы.

1 Выбирается из ErWin Toolbox тип сущности зависимая (если она потомок) или независимая (если она родитель).

2 Щелкнуть по месту расположения сущности на рабочем поле.

3 Выбрать в Toolbox стрелку Select. Дважды щелкнуть по полю сущности. В открывшимся редакторе Entity- Attribute Editor в окне Entity записать имя сущности. OK.

Задача 2. Проектирование отношений.

Цель: проектирование и осмысление характера взаимосвязей между сущностями.

Последовательность действий:

1. определить тип отношения;
2. указать имя отношения.

Требования к диаграмме:

- имена связей должны выбираться в глагольной форме, так, чтобы диаграмма читалась осмысленными фразами русского языка;
- при разработке допускаются определенные, неопределенные, связи типа “many-many” и категоризационные связи.

Порядок проектирования диаграммы.

1 Устанавливаются в поле экрана сущности Родителя и Потомка. Щелкнуть по полю тип отношений ErWin Toolbox (определенное, неопределенное, неспецифическое).

2 Щелкнуть сначала по полю сущности Родителя, а затем по полю сущности Потомка.

3 Дважды щелкнуть по отношению. В открывшимся редакторе отношений в окне Verb Phrase вводят глагольную форму, связывающую обе сущности в смысловое предложение. Триггерами Кардинальности указывается мощность отношения (P, Z или число).

Обычно ставится максимальная. В дисплейном окне Rolename (можно изменить ролевое назначение ключа). В окне Foreign Key указывается (название ключа наследования).

4 Чтобы наблюдать на рабочем экране имя отношения необходимо выбрать в панели меню DISPLAY/ Verb Phrase.

5 В случае необходимости, чтобы добавить в схему отношение типа категории между Родителем и двумя Потомками необходимо:

- выбрать тип категории (категория с одной линией показывает, что имеются другие сущности категории, которые не включены в схему, категория с двумя линиями указывает, что все категории учтены);
- щелкнуть последовательно сначала по сущности Родителя, а затем по сущности одного из Потомков;
- щелкнуть по месту установки знака отношения категории, а затем щелкнуть по оставшейся сущности.

Задача 3. Проектирование КВ диаграммы.

Цель: проектирование идентификаторов сущностей и уяснение логики взаимосвязей на уровне идентификаторов.

Последовательность действий:

- преобразовать все неспецифические отношения в специфические;
- определить возможные ключи независимых сущностей и выделить первичные ключи;
- ввести формальные определения имен ключевых атрибутов в глоссарий;
- показать первичные и все возможные ключи на диаграмме;

Требования к диаграмме:

- на диаграмме допускаются только специфические и категоризационные связи;
- глоссарий должен содержать формальные определения ключей.

Физическое проектирование.

Задача 4. Физическое проектирование.

Цель: разработка структуры реляционной базы данных для выбраннной СУБД.

Последовательность действий:

1. поставить в соответствие именам сущностей и атрибутов логической модели имена таблиц и полей БД (физические имена);
2. разработать сопроводительную документацию.

Требования к диаграмме:

1. Проектируются типы данных в соответствующем СУБД. Для этого включается в верхней линейке пиктограмма PHYSICAL VIEW,
2. Выбирается указателем мышки проектируемая сущность, нажатой правой выбирают Вариант СУБД DATABASE, в редакторе выбирают тип данных,
3. размер записи, ОК.

4. Описываются свойства связей между таблицами, в частности, реакцию сервера на попытки нарушения ссылочной целостности со стороны клиентского приложения. Для этого выделив связь, щелкнуть правой кнопкой и выбрать в редакторе отношений опцию RELATIONSHIP INTEGRITY, при этом можно отредактировать ее VERB PHRASE, установить триггеры запрещения действий клиентскому приложению по удалению записей CHILD DELETE- NONE, PARENT DELETE- NONE, а все остальное - RESTRICT.

После этого, можно создавать на сервере схему БД. Для этого отредактировать тип данных (войти в какую нибудь сущность и правой кнопкой щелкнуть, а затем выбрать Attribute Editors, ввести редактируемую сущность в окне Entity), введя типы для атрибутов и ключей. Ввести определения атрибутов и имена владельца данных. Вызвать Domain Editor (Column Propery Editor/ Domain). Ввести имена доменов и присвоить им типы (длина) и определения.

Задание

1. Провести системный анализ предметной области согласно вашему варианту.
2. Разработать концептуальную модель данных.
3. Разработать в логическую модель данных.
4. Разработать физическую модель данных.

При разработке можно использовать CASE-систему ErWin Data Modeler (или аналогичную).

Полученный набор отношений должен находиться в третьей или выше нормальной форме и содержать не менее 5-ти сущностей.

Контрольные вопросы

1. В чем разница между концептуальной, логической и физической моделями данных?

2. Объясните смысл терминов (реляционная модель данных, мощность отношения, тип отношения).

3. Дайте определения нормальных форм.

4. Объясните применение ограничительных условий, поддерживающих целостность данных.

Содержание и оформление отчета

Отчет должен содержать: титульный лист, название и цель работы; вариант задания; скриншоты результатов работы; выводы по работе.

Лабораторная работа №7 (3)

Тема: Создание баз данных и использование операторов манипулирования данными в microsoft sql server. (Можно скачать с http://www.microsoft.com/ru-ru/download/details.aspx?id=43351)

Цель: С помощью операторов языка Transact SQL научиться создавать базы данных и совокупность связанных таблиц, принадлежащих указанной базе данных, использовать операторы манипулирования данными Select, Insert, Update, Delete.

Содержание работы:

1. Познакомиться с набором утилит, входящих в состав MS SQL Server.
2. Познакомиться с работой утилиты SQL Server Management Studio. (http://www.microsoft.com/ru-RU/download/details.aspx?id=22985)
3. Создать с помощью приведенных операторов пример базы данных «Книжное дело».
4. По выданным вариантам создать персональную базу данных с набором связанных таблиц.
5. Выполнить запросы к БД «Книжное дело» по выданным вариантам.

Пояснения к выполнению работы

В качестве примера базы данных, которая будет создана программно с помощью операторов языка Transact SQL, выберем БД «Книжное дело» (рис. 1). Структура таблиц данной БД представлена в табл. 1-5.

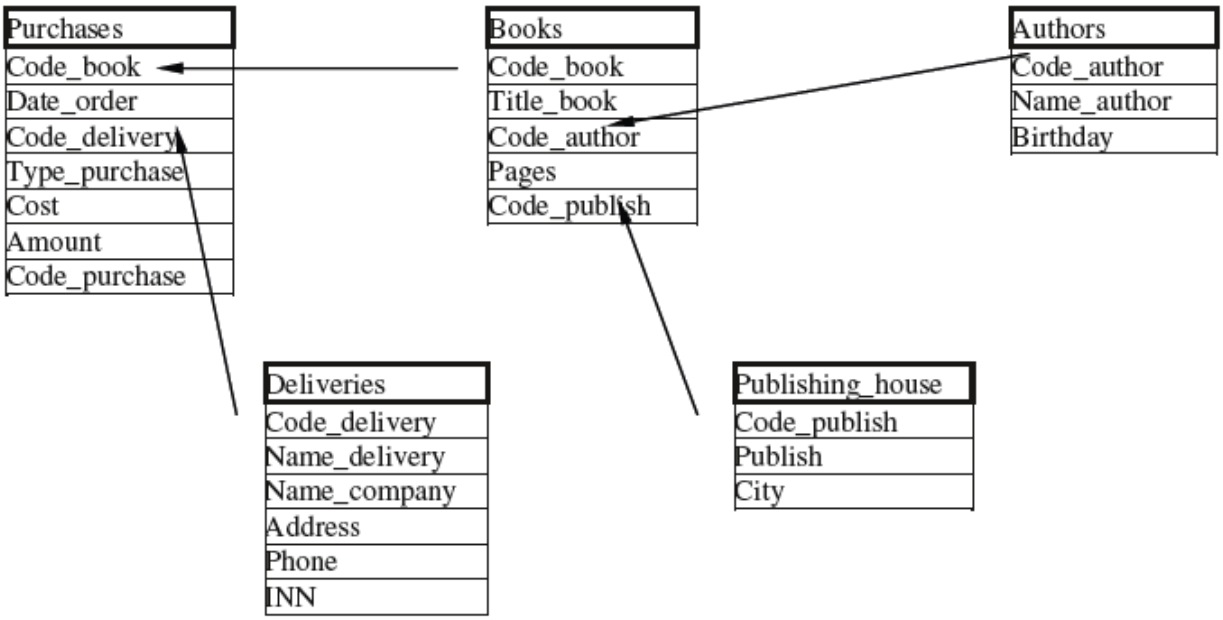


Рис. 1. Фрагмент базы данных «Книжное дело»

Таблица 1

Покупки (название таблицы Purchases)

Название поля	Тип поля	Описание поля
Code_book	Int	Код закупаемой книги
Date_order	DateTime	Дата заказа книги
Code_delivery	Int	Код поставщика
Type_purchase	Bit	Тип закупки (опт/ розница)
Cost	Money	Стоимость единицы товара
Amount	Int	Количество экземпляров
Code_purchase	Int	Код покупки

Таблица 2

Справочник книг (название таблицы Books)

Название поля	Тип поля	Описание поля
Code_book	Int	Код книги
Title_book	Char	Название книги
Code_author	Int	Код автора
Pages	Int	Количество страниц
Code_publish	Int	Код издательства

Таблица 3

Справочник авторов (название таблицы Authors)

Название поля	Тип поля	Описание поля
Code_author	Int	Код автора
Name_author	Char	Фамилия, имя, отчество автора
Birthday	DateTime	Дата рождения

Таблица 4

Справочник поставщиков (название таблицы Deliveries)

Название поля	Тип поля	Описание поля
Code_delivery	Int	Код поставщика
Name_delivery	Char	Фамилия, и., о. ответственного лица
Name_company	Char	Название компании-поставщика
Address	Char	Юридический адрес
Phone	Numeric	Телефон контактный
INN	Char	ИНН

Таблица 5

Справочник издательств (название таблицы Publishing_house)

Название поля	Тип поля	Описание поля
Code_publish	Int	Код издательства
Publish	Char	Издательство
City	Char	Город

Запустить **SQL Server Management Studio**, проверить включение сервера.

Для написания программного кода в **SQL Server Management Studio** нужно нажать кнопку «Создать запрос» («New query») на панели инструментов «Стандартная» («Standart»).

Создать новую базу данных с названием **DB_Books** с помощью команды:

```
CREATE DATABASE DB_BOOKS
```

Для выполнения команды нажать F5.

Открыть утилиту **SQL Server Management Studio**. Проверить наличие БД DB_Books, если ее не видно в разделе DataBases, то нажать F5 для обновления.

Создать в ней перечисленные таблицы с помощью следующих команд (для создания новой страницы для кода в SQL Server Management Studio нажать кнопку «Создать запрос»):

```
use DB_BOOKS
CREATE TABLE Authors(Code_author INT PRIMARY KEY, name_author CHAR(30), Birthday DATETIME)
CREATE TABLE Publishing_house(Code_publish INT PRIMARY KEY, Publish CHAR(30), City CHAR(20))
CREATE TABLE Books(Code_book INT PRIMARY KEY, Title_book CHAR(40), Code_author INT FOREIGN KEY REFERENCES Authors(Code_author), Pages INT, Code_publish INT FOREIGN KEY REFERENCES Publishing_house(Code_publish))
CREATE TABLE Deliveries(Code_delivery INT PRIMARY KEY, Name_delivery CHAR(30), Name_company CHAR(20), Address VARCHAR(100), Phone BIGINT, INN CHAR(13))
CREATE TABLE Purchases(Code_purchase INT PRIMARY KEY, Code_book INT FOREIGN KEY REFERENCES Books(Code_book), Date_order SMALLDATETIME, Code_delivery INT FOREIGN KEY REFERENCES Deliveries(Code_delivery), Type_purchase BIT, Cost FLOAT, Amount INT)
```

Запустить команду клавишей F5.

В утилите SQL Server Management Studio проверить наличие БД DB_Books и таблиц в ней.

В разделе диаграмм создать новую диаграмму, в которую добавить из списка пять наших таблиц, проверить связи между таблицами.

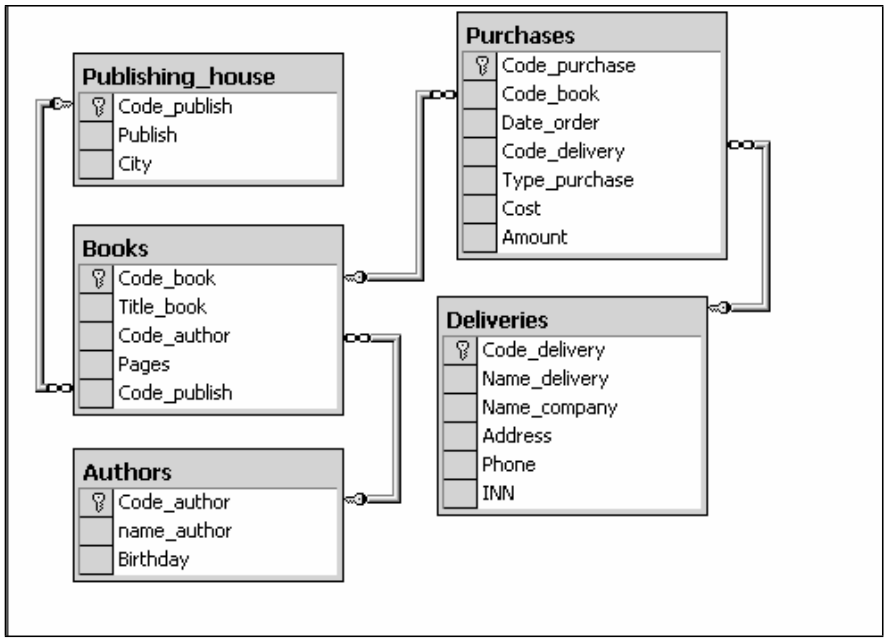


Рис. 1.3. Результат создания диаграммы

Таблица 6. Соответствие типов данных Microsoft Access и Microsoft SQL

№	Тип данных Microsoft Access	Тип данных Microsoft SQL	Описание типа данных Microsoft SQL
1	Текстовый	nvarchar	Тип данных для хранения текста до 4000 символов
2	Поле MEMO	ntext	Тип данных для хранения символов в кодировке Unicode до 1 073 741 823 символов
3	Числовой	int	Численные значения (целые) в диапазоне от -2 147 483 648 до +2 147 483 647
4	Дата/время	smalldatetime	Дата и время от 1 января 1900 г. до 6 июня 2079 года с точностью до одной минуты
5	Денежный	money	Денежный тип данных, значения которого лежат в диапазоне от -922 337 203 685 477.5808 до +922 337 203 685 477.5807, с точностью до одной десятитысячной
6	Счетчик	int	См. пункт 3
7	Логический	bit	Переменная, способная принимать только два значения - 0 или 1
8	Поле объекта OLE	image	Переменная для хранения массива байтов от 0 до 2 147 483 647 байт
9	Гиперссылка	ntext	См. пункт 2
10	Мастер подстановок	nvarchar	См. пункт 1

Варианты заданий

1. Создать новую базу данных по своей предметной области, создать в ней перечисленные таблицы. Для этого в утилите **SQL Server Management**

Studio создать отдельные скрипты по каждому запросу В сами скрипты копировать текст задания в виде комментария. Можно сохранять все выполненные запросы в одном файле.

2. В созданной базе данных **DB_Books** создать отдельные скрипты по каждому запросу. В сами скрипты копировать текст задания в виде комментария. Можно сохранять все выполненные запросы в одном файле. Для проверки работы операторов SELECT предварительно создайте скрипт, который с помощью операторов INSERT заполнит все таблицы БД DB_Books несколькими записями.

Список вариантов заданий

Вариант	Список номеров упражнений												
1	1	6	11	16	21	26	31	36	41	46	51	56	61
2	2	7	12	17	22	27	32	37	42	47	52	57	62
3	3	8	13	18	23	28	33	38	43	48	53	58	63
4	4	9	14	19	24	29	34	39	44	49	54	59	64
5	5	10	15	20	25	30	35	40	45	50	55	60	65
6	6	11	16	21	26	31	36	41	46	51	56	61	1
7	7	12	17	22	27	32	37	42	47	52	57	62	2
8	8	13	18	23	28	33	38	43	48	53	58	63	3
9	9	14	19	24	29	34	39	44	49	54	59	64	4
10	10	15	20	25	30	35	40	45	50	55	60	65	5
11	2	6	12	16	22	26	32	36	42	46	52	56	62
12	1	5	11	15	21	25	31	35	41	45	51	55	61
13	3	7	13	17	23	27	33	37	43	47	53	57	63

Сортировка

1. Выбрать все сведения о книгах из таблицы Books и отсортировать результат по коду книги (поле Code_book).
2. Выбрать из таблицы Books коды книг, названия и количество страниц (поля Code_book, Title_book и Pages), отсортировать результат по названиям книг (поле Title_book по возрастанию) и по полю Pages (по убыванию).
3. Выбрать из таблицы Deliveries список поставщиков (поля Name_delivery, Phone и INN), отсортировать результат по полю INN (по убыванию).

Изменение порядка следования полей

4. Выбрать все поля из таблицы Deliveries таким образом, чтобы в результате порядок столбцов был следующим: Name_delivery, INN, Phone, Address, Code_delivery.

5. Выбрать все поля из таблицы Publishing_house таким образом, чтобы в результате порядок столбцов был следующим: Publish, City, Code_publish.

Выбор некоторых полей из двух таблиц

6. Выбрать из таблицы Books названия книг и количество страниц (поля Title_book и Pages), а из таблицы Authors выбрать имя соответствующего автора книги (поле Name_ author).

7. Выбрать из таблицы Books названия книг и количество страниц (поля Title_book и Pages), а из таблицы Deliveries выбрать имя соответствующего поставщика книги (поле Name_delivery).

8. Выбрать из таблицы Books названия книг и количество страниц (поля Title_book и Pages), а из таблицы Publishing_house выбрать название соответствующего издательства и места издания (поля Publish и City).

Условие неточного совпадения

9. Выбрать из справочника поставщиков (таблица Deliveries) названия компаний, телефоны и ИНН (поля Name_company, Phone и INN), у которых название компании (поле Name_company) начинается с ‘ОАО’.

10. Выбрать из таблицы Books названия книг и количество страниц (поля Title_book и Pages), а из таблицы Authors выбрать имя соответствующего автора книг (поле Name_ author), у которых название книги начинается со слова ‘Мемуары’.

11. Выбрать из таблицы Authors фамилии, имена, отчества авторов (поле Name_ author), значения которых начинаются с ‘Иванов’.

Точное несовпадение значений одного из полей

12. Вывести список названий издательств (поле Publish) из таблицы Publishing_house, которые не находятся в городе ‘Москва’ (условие по полю City).

13. Вывести список названий книг (поле Title_book) из таблицы Books, которые выпущены любыми издательствами, кроме издательства ‘Питер-Софт’ (поле Publish из таблицы Publishing_house).

Выбор записей по диапазону значений (Between)

14. Вывести фамилии, имена, отчества авторов (поле Name_author) из таблицы Authors, у которых дата рождения (поле Birthday) находится в диапазоне 01.01.1840 – 01.06.1860.

15. Вывести список названий книг (поле Title_book из таблицы Books) и количество экземпляров (поле Amount из таблицы Purchases), которые были закуплены в период с 12.03.2003 по 15.06.2003 (условие по полю

Date_order из таблицы Purchases).

16. Вывести список названий книг (поле Title_book) и количество страниц (поле Pages) из таблицы Books, у которых объем в страницах укладывается в диапазон 200 – 300 (условие по полю Pages).

17. Вывести список фамилий, имен, отчеств авторов (поле Name_author) из таблицы Authors, у которых фамилия начинается на одну из букв диапазона ‘В’ – ‘Г’ (условие по полю Name_author).

Выбор записей по диапазону значений (In)

18. Вывести список названий книг (поле Title_book из таблицы Books) и количество (поле Amount из таблицы Purchases), которые были поставлены поставщиками с кодами 3, 7, 9, 11 (условие по полю Code_delivery из таблицы Purchases).

19. Вывести список названий книг (поле Title_book) из таблицы Books, которые выпущены следующими издательствами: ‘Питер-Софт’, ‘Альфа’, ‘Наука’ (условие по полю Publish из таблицы Publishing_house).

20. Вывести список названий книг (поле Title_book) из таблицы Books, которые написаны следующими авторами: ‘Толстой Л.Н.’, ‘Достоевский Ф.М.’, ‘Пушкин А.С.’ (условие по полю Name_author из таблицы Authors).

Выбор записей с использованием Like

21. Вывести список авторов (поле Name_author) из таблицы Authors, которые начинаются на букву ‘К’.

22. Вывести названия издательств (поле Publish) из таблицы Publishing_house, которые содержат в названии сочетание ‘софт’.

23. Выбрать названия компаний (поле Name_company) из таблицы Deliveries, у которых значение оканчивается на ‘ский’.

Выбор записей по нескольким условиям

24. Выбрать коды поставщиков (поле Code_delivery), даты заказов (поле Date_order) и названия книг (поле Title_book), если количество книг (поле Amount) в заказе больше 100 или цена (поле Cost) за книгу находится в диапазоне от 200 до 500.

25. Выбрать коды авторов (поле Code_author), имена авторов (поле Name_author), названия соответствующих книг (поле Title_book), если код издательства (поле Code_Publish) находится в диапазоне от 10 до 25 и количество страниц (поле Pages) в книге больше 120.

26. Вывести список издательств (поле Publish) из таблицы Publishing_house, в которых выпущены книги, названия которых (поле Title_book) начинаются со слова ‘Труды’ и город издания (поле City) – ‘Новосибирск’.

Многотабличные запросы (выборка из двух таблиц, выборка из трех таблиц с использованием JOIN)

27. Вывести список названий компаний-поставщиков (поле Name_company) и названия книг (поле Title_book), которые они поставили в период с 01.01.2002 по 31.12.2003 (условие по полю Date_order).

28. Вывести список авторов (поле Name_author), книги которых были выпущены в издательстве ‘Мир’ (условие по полю Publish).

29. Вывести список поставщиков (поле Name_company), которые поставляют книги издательства ‘Питер’ (условие по полю Publish).

30. Вывести список авторов (поле Name_author) и названия книг (поле Title_book), которые были поставлены поставщиком ‘ОАО Книготорг’ (условие по полю Name_company).

Вычисления

31. Вывести суммарную стоимость партии одноименных книг (использовать поля Amount и Cost) и название книги (поле Title_book) в каждой поставке.

32. Вывести стоимость одной печатной страницы каждой книги (использовать поля Cost и Pages) и названия соответствующих книг (поле Title_book).

33. Вывести количество лет с момента рождения авторов (использовать поле Birthday) и имена соответствующих авторов (поле Name_author).

Вычисление итоговых значений с использованием агрегатных функций

34. Вывести общую сумму поставок книг (использовать поле Cost), выполненных ‘ЗАО Оптторг’ (условие по полю Name_company).

35. Вывести общее количество всех поставок (использовать любое поле из таблицы Purchases), выполненных в период с 01.01.2003 по 01.02.2003 (условие по полю Date_order).

36. Вывести среднюю стоимость (использовать поле Cost) и среднее количество экземпляров книг (использовать поле Amount) в одной поставке, где автором книги является ‘Акунин’ (условие по полю Name_author).

37. Вывести все сведения о поставке (все поля таблицы Purchases), а также название книги (поле Title_book) с минимальной общей стоимостью (использовать поля Cost и Amount).

38. Вывести все сведения о поставке (все поля таблицы Purchases), а также название книги (поле Title_book) с максимальной общей стоимостью (использовать поля Cost и Amount).

Изменение наименований полей

39. Вывести название книги (поле Title_book), суммарную стоимость партии одноименных книг (использовать поля Amount и Cost), поместив в результат в поле с названием Itogo, в поставках за период с 01.01.2002 по

01.06.2002 (условие по полю Date_order).

40. Вывести стоимость одной печатной страницы каждой книги (использовать поля Cost и Pages), поместив результат в поле с названием One_page, и названия соответствующих книг (поле Title_book).

41. Вывести общую сумму поставок книг (использовать поле Cost) и поместить результат в поле с названием Sum_cost, выполненных ‘ОАО Луч’ (условие по полю Name_company).

Использование переменных в условии

42. Вывести список сделок (все поля из таблицы Purchases) за последний месяц (условие с использованием поля Date_order).

43. Вывести список авторов (поле Name_author), возраст которых меньше заданного пользователем (условие с использованием поля Birthday).

44. Вывести список книг (поле Title_book), которых закуплено меньше, чем указано в запросе пользователя (условие с использованием поля Amount).

Использование переменных вместо названий таблиц

45. Вывести список названий компаний-поставщиков (поле Name_company) и названия книг (поле Title_book), которые они поставили.

46. Вывести список авторов (поле Name_author), книги которых были выпущены в издательствах ‘Мир’, ‘Питер Софт’, ‘Наука’ (условие по полю Publish).

47. Вывести список издательств (поле Name_company), книги которых были поставлены по цене 150 руб. (поле Cost).

Выбор результата

48. Вывести список названий книг (поле Title_book) и количества страниц (поле Pages) в каждой книге.

49. Вывести список названий компаний-поставщиков (поле Name_company).

50. Вывести список авторов (поле Name_author).

Использование функций совместно с подзапросом

51. Вывести список книг (поле Title_book), у которых количество страниц (поле Pages) больше среднего количества страниц всех книг в таблице.

52. Вывести список авторов (поле Name_author), возраст которых меньше среднего возраста всех авторов в таблице (условие по полю Birthday).

53. Вывести список книг (поле Title_book), у которых количество страниц (поле Pages) равно минимальному количеству страниц книг, представленных в таблице.

Использование квантора существования в запросах

54. Вывести список издательств (поле Publish), книги которых были приобретены оптом (‘опт’ из поля Type_Purchase).
55. Вывести список авторов (поле Name_author), книг которых нет в таблице Books.
56. Вывести список книг (поле Title_book), которые были поставлены поставщиком ‘ЗАО Квантор’ (условие по полю Name_company).

Оператор обработки данных Update

57. Изменить в таблице Books содержимое поля Pages на 300, если код автора (поле Code_author) =56 и название книги (поле Title_book) =’Мемуары’.
58. Изменить в таблице Deliveries содержимое поля Address на ‘нет сведений’, если значение поля является пустым.
59. Увеличить в таблице Purchases цену (поле Cost) на 20 процентов, если заказы были оформлены в течение последнего месяца (условие по полю Date_order).

Оператор обработки данных Insert

60. Добавить в таблицу Purchases новую запись, причем так, чтобы код покупки (поле Code_purchase) был автоматически увеличен на единицу, а в тип закупки (поле Type_purchase) внести значение ‘опт’.
61. Добавить в таблицу Books новую запись, причем вместо ключевого поля поставить код (поле Code_book), автоматически увеличенный на единицу от максимального кода в таблице, вместо названия книги (поле Title_book) написать ‘Наука. Техника. Инновации’.
62. Добавить в таблицу Publish_house новую запись, причем вместо ключевого поля поставить код (поле Code_publish), автоматически увеличенный на единицу от максимального кода в таблице, вместо названия города – ‘Москва’ (поле City), вместо издательства – ‘Наука’ (поле Publish).

Оператор обработки данных Delete

63. Удалить из таблицы Purchases все записи, у которых количество книг в заказе (поле Amount) = 0.
64. Удалить из таблицы Authors все записи, у которых нет имени автора в поле Name_Author.
65. Удалить из таблицы Deliveries все записи, у которых не указан ИНН (поле INN пустое).

Контрольные вопросы.

1. Для чего используется SQL Server Management Studio? Можно ли добиться тех же целей без использования данной утилиты?
2. Опишите операцию подключения к серверу.
3. Перечислите операторы, используемые при создании базы данных с объяснением их действия.
4. Перечислите операторы, используемые при создании таблиц с объяснением их действия.
5. Как обратиться к нужной базе данных для выполнения действий с ней?
6. Необходимо ли сохранять коды запросов и почему?
7. Как в Microsoft SQL сервере создать поле-счетчик?

Лабораторная работа №8(4)

ОСВОЕНИЕ ПРОГРАММИРОВАНИЯ С ПОМОЩЬЮ ВСТРОЕННОГО ЯЗЫКА TRANSACT SQL В MICROSOFT SQL SERVER

Цель работы – знакомство с основными принципами программирования в MS SQL Server средствами встроенного языка Transact SQL.

Содержание работы:

1. Знакомство с правилами обозначения синтаксиса команд в справочной системе MS SQL Server.
2. Изучение правил написания программ на Transact SQL.
3. Изучение правил построения идентификаторов, правил объявления переменных и их типов.
4. Изучение работы с циклами и ветвлениями.
5. Изучение работы с переменными типа Table и Cursor.
6. Проработка всех примеров, анализ результатов их выполнения.
7. Выполнение индивидуальных заданий по вариантам.

Пояснения к выполнению работы

Для освоения программирования используем пример базы данных с названием **DB_Books**. При выполнении примеров и заданий обращайте внимание на соответствие названий БД, таблиц и других объектов проекта.

Специальные знаки и простейшие операторы в Transact SQL

Знак	Назначение	Знак	Назначение
*	Знак умножения	" "	В них заключают строковые зна чения, если SET QUOTED_IDENTIFIER OFF
-	Знак вычитания	‘ ’	В них заключают строковые зна чения
%	Остаток от деления двух чисел	<>	Не равно
+	Знак сложения или конкатена ции (объединение двух строк в одну)	[ ]	Аналог кавычек, в них можно заключать названия идентифика торов, если в их названиях встречаются пробелы
=	Знак равенства или сравнения	!<	Не менее чем
<=	Меньше или равно	!>	Не более чем
>=	Больше или равно	>	Больше
!=	Не равно	<	Меньше
@	Ставится перед именем пере менной	.	Разделяет родительские и подчи ненные объекты
@@	Указывает на системные функции	/	Знак деления
--	Однострочный комментарий или комментарий с текущей позиции и до конца строки	/* */	Многострочный комментарий

Идентификаторы – это имена объектов, на которые можно ссылаться в программе, написанной на языке Transact SQL. Первый символ может со стоять из букв английского алфавита или “_”, “@”, “#”. Остальные дополнительно из цифр и «$».

Имя идентификатора не должно совпадать с зарезервированным словом.

Для ограничителей идентификаторов при установленном параметре

SET QUOTED_IDENTIFIER ON

можно использовать как квадратные скобки, так и одинарные кавычки, а строковые значения только в одинарных кавычках (режим по умолчанию).

Если использовать установленный параметр в режиме

SET QUOTED_IDENTIFIER OFF,

то в качестве ограничителей идентификаторов можно использовать только квадратные скобки, а строковые значения указываются в одинарных или двойных кавычках.

Переменные используются для сохранения промежуточных данных в хранимых процедурах и функциях. Все переменные считаются локальными. Имя переменной должно начинаться с @.

Объявление переменных

Синтаксис в обозначениях MS SQL Server:

DECLARE @имя_переменной_1 тип_переменной, …, @имя_переменной_N тип_переменной

Если тип переменной предполагает указание размера, то используется следующий синтаксис для объявления переменных:

DECLARE @имя_переменной_1 тип_переменной (размер), …, @имя_переменной_N тип_переменной(размер)

Пример:

```
DECLARE @a INT, @b NUMERIC(10,2)
DECLARE @str CHAR(20)
```

Присвоение значений переменным и вывод значений на экран

Присвоение с помощью **SET** – обычное присвоение, синтаксис: SET @имя_переменной = значение.

Пример:

```
DECLARE @a INT, @b NUMERIC(10,2) SET @a = 20
SET @b = (@a+@a)/15
SELECT @b --вывод на экран результата
```

Присвоение с помощью **SELECT** – помещение результата запроса в переменную. Если в результате выполнения запроса не будет возвращено ни одной строки, то значение переменной не меняется, т.е. остается старым.

Пример:

```
DECLARE @a INT
SELECT @a = COUNT(*) FROM Authors
```

Пример:

```
DECLARE @str CHAR(30)
```

```
SELECT @str = name FROM Authors
```

В данном примере в переменную поместится последнее значение из результата запроса.

Сочетание ключевых слов SET и SELECT

Пример:

```
DECLARE @a INT
SET @a = (SELECT COUNT(*) FROM Authors)
```

Работа с датой и временем

Оператор SET DATEFORMAT dmy | ymd | mdy задает порядок следо вания компонентов даты.

Пример:

```
SET DATEFORMAT dmy
DECLARE @d DateTime
SET @d = '31.01.2005 13:23:15'
SET @d = @d+1 SELECT @d
```

Создание временной таблицы через переменную типа TABLE

Объявляется через DECLARE с указанием в скобках столбцов табли цы, их типов, размеров, значений по умолчанию, а также индексов типа PRIMARY KEY или UNIQUE.

Пример:

```
DECLARE @mytable TABLE(id INT, myname CHAR(20) DEFAULT 'Введите имя')
INSERT INTO @mytable(id) VALUES (1) SELECT * FROM @mytable
```

Пример:

```
DECLARE @mytable TABLE(id INT, myname CHAR(20) DEFAULT 'Введите имя')
INSERT @mytable SELECT Code_publish, City FROM Publishing_house SELECT
* FROM @mytable
```

Преобразование типов переменных

Функция **CAST** возвращает значение, преобразованное к указанному типу:
CAST(@переменная или значение AS требуемый_тип_данных)

Пример:

```
DECLARE @d DateTime, @str CHAR(20)
SET @d = '31.01.2005 13:23:15'
SET @str = CAST(@d AS CHAR(20))
SELECT 2str
```

Функция **CONVERT** возвращает значение, преобразованное к указан ному типу по заданному формату. Изучить дополнительно, по желанию.

Операторские скобки

```
BEGIN
```

```
/* в них нельзя помещать команды, изменяющие структуры объектов БД.
Операторские скобки должны содержать хотя бы один оператор. Требуются
для конструкций поливариантных ветвлений, условных и циклических
конструкций
*/
END
```

Условная конструкция IF

Синтаксис:

IF условие

Набор операторов1

ELSE

Набор операторов2

Пример:

```
DECLARE @a INT DECLARE @str CHAR(30)
SET @a = (SELECT COUNT(*) FROM Authors)
IF @a >10
BEGIN
 SET @str = 'Количество авторов больше 10'
 SELECT @str
END
ELSE
BEGIN
 SET @str = 'Количество авторов = ' + str(@a)
 SELECT @str
END
```

Цикл WHILE

Синтаксис: WHILE *Условие*

Набор операторов1

BREAK

Набор опреторов2

CONTINUE

Конструкции BREAK и CONTINUE являются необязательными.

Цикл можно принудительно остановить, если в его теле выполнить команду BREAK. Если же нужно начать цикл заново, не дожидаясь выпол нения всех команд в теле, необходимо выполнить команду CONTINUE.

Пример:

```
DECLARE @a INT
SET @a = 1
WHILE @a <100
 BEGIN
  PRINT @a - вывод на экран значения переменной
  IF (@a>40) AND (@a<50)
   BREAK --выход и выполнение 1-й команды за циклом
  ELSE

   SET @a = @a+rand()*10
```

```
    CONTINUE
 END
PRINT @a
```

Объявление курсора

CURSOR – это набор строк, являющийся результатом выполнения за проса. В один момент времени доступна лишь одна строка (текущая), по курсору можно передвигаться и получать доступ к элементарным данным. При объявлении курсора создается временная копия данных, которая сохраняется в БД tempdb.
Динамический курсор – данные в курсоре могут быть изменены.
Статический курсор – данные в курсоре не меняются.

Стандартный способ объявления курсора, синтаксис в обозначениях
DECLARE cursor_name [INSENSITIVE] [SCROLL] CURSOR
FOR select_statement
[FOR { READ ONLY | UPDATE [OF column_name [,...n]] }]

Примеры объявления курсоров:

```
DECLARE MyCursor1 CURSOR FOR (SELECT * FROM Authors)
/*объявили курсор с названием MyCursor1, который содержит всю
информацию об авторах, двигаться по нему можно только от первой записи
вниз до последней. Курсор является динамическим.*/

DECLARE MyCursor1 INSENSITIVE CURSOR FOR (SELECT * FROM Authors)
/*объявили курсор с названием MyCursor1, который содержит всю
информацию об авторах, двигаться по нему можно только от первой записи
вниз до последней. Курсор является статическим.*/

DECLARE MyCursor1 SCROLL CURSOR FOR (SELECT * FROM Authors)
/*объявили курсор с названием MyCursor1, который содержит всю
информацию об авторах, двигаться по нему можно в любом направлении.
Курсор является динамическим.*/

DECLARE MyCursor1 INSENSITIVE SCROLL CURSOR FOR (SELECT * FROM
Authors)
/*объявили курсор с названием MyCursor1, который содержит всю
информацию об авторах, двигаться по нему можно в любом направлении.
Курсор является статическим.*/

DECLARE MyCursor1 CURSOR FOR (SELECT * FROM Authors) FOR READ ONLY
/*объявили курсор с названием MyCursor1, который содержит всю
информацию об авторах, двигаться по нему можно только от первой записи
вниз до последней. Курсор является динамическим. Данные доступны
только для чтения.*/

DECLARE MyCursor1 CURSOR FOR (SELECT * FROM Authors) FOR UPDATE
/*объявили курсор с названием MyCursor1, который содержит всю
информацию об авторах, двигаться по нему можно только от первой записи
вниз до последней. Курсор является динамическим. Данные курсора можно
менять.*/
```

Операторы для работы с курсором

Прежде чем обратиться к данным курсора, его нужно после объявления открыть.

Синтаксис оператора OPEN в обозначениях MS SQL Server: OPEN { { [GLOBAL] *cursor_name* } | *cursor_variable_name* } **Пример:**

```
DECLARE MyCursor1 CURSOR FOR (SELECT * FROM Authors)
OPEN MyCursor1
```

После прекращения работы с курсором, его нужно закрыть. Курсор остается доступным для последующего использования в рамках процедуры или триггера, в котором он создан.

Синтаксис оператора CLOSE в обозначениях MS SQL Server: CLOSE { { [GLOBAL] *cursor_name* } | *cursor_variable_name* } **Пример:**

```
DECLARE MyCursor1 CURSOR FOR (SELECT * FROM Authors)
OPEN MyCursor1
--здесь операторы работы с курсором
CLOSE MyCursor1
```

Если курсором больше не будут пользоваться, то его необходимо уничтожить и освободить переменную.

Синтаксис оператора DEALLOCATE в обозначениях MS SQL Server: DEALLOCATE { { [GLOBAL] *cursor_name* } | *@cursor_variable_name* } **Пример:**

```
DECLARE MyCursor1 CURSOR FOR (SELECT * FROM Authors)
OPEN MyCursor1
--здесь операторы работы с курсором
CLOSE MyCursor1
DEALLOCATE MyCursor1
```

FETCH – оператор движения по записям курсора и извлечения данных те кущей записи в указанные переменные.

Синтаксис оператора FETCH в обозначениях MS SQL Server: FETCH

[[NEXT | PRIOR | FIRST | LAST
| ABSOLUTE { *n* | *@nvar* }
| RELATIVE { *n* | *@nvar* }
] FROM
]
{ { [GLOBAL] *cursor_name* } | *@cursor_variable_name* } [INTO *@variable_name* [**,***...n*]]

Пример:

```
DECLARE MyCursor1 SCROLL CURSOR FOR (SELECT * FROM Authors)
DECLARE @i BIGINT, @s CHAR(20), @d smalldatetime
OPEN MyCursor1
FETCH FIRST FROM MyCursor1 INTO @i, @s, @d
PRINT @i
PRINT @s
PRINT @d
CLOSE MyCursor1
DEALLOCATE MyCursor1
```

@@FETCH_STATUS – данная функция определяет признак конца или начала текущего курсора. Функция принимает одно из следующих значений: 0 – находимся в пределах

курсора, не в конце; 1 – попытка выйти за пределы первой записи вверх (в никуда); 2 – попытка выйти за пределы последней записи вниз (в никуда).

Пример:

```
DECLARE MyCursor1 SCROLL CURSOR FOR (SELECT * FROM Authors)
DECLARE @i BIGINT, @s CHAR(20), @d smalldatetime
OPEN MyCursor1
FETCH FIRST FROM MyCursor1 INTO @i, @s, @d
WHILE @@FETCH_STATUS = 0
  BEGIN
   FETCH NEXT FROM MyCursor1 INTO @i, @s, @d
   PRINT @i
   PRINT @s
   PRINT @d
  END
CLOSE MyCursor1
DEALLOCATE MyCursor1
```

Встроенные функции

Встроенные функции, имеющиеся в распоряжении пользователей при работе с SQL, можно условно разделить на следующие группы:

- математические функции;
- строковые функции;
- функции для работы с датой и временем;
- функции конфигурирования;
- функции системы безопасности;
- функции управления метаданными;
- статистические функции.

Использование функций для работы со строковыми переменными

Краткий обзор строковых функций

Название функции	**Действие, выполняемое функцией**
ASCII	Возвращает код ASCII левого символа строки
CHAR	По коду ASCII возвращает символ
CHARINDEX	Определяет порядковый номер символа, с которого начинается вхождение подстроки в строку
DIFFERENCE	Возвращает показатель совпадения строк
LEFT	Возвращает указанное число символов с начала строки
LEN	Возвращает длину строки
LOWER	Переводит все символы строки в нижний регистр
LTRIM	Удаляет пробелы в начале строки
NCHAR	Возвращает по коду символ Unicode
PATINDEX	Выполняет поиск подстроки в строке по указанному шаблону
REPLACE	Заменяет вхождения подстроки на указанное значение
QUOTENAME	Конвертирует строку в формат Unicode
REPLICATE	Выполняет тиражирование строки определенное число раз
REVERSE	Возвращает строку, символы которой записаны в обратном поряд ке
RIGHT	Возвращает указанное число символов с конца строки

RTRIM	Удаляет пробелы в конце строки
SOUNDEX	Возвращает код звучания строки
SPACE	Возвращает указанное число пробелов
STR	Выполняет конвертирование значения числового типа в символь ный формат
STUFF	Удаляет указанное число символов, заменяя новой подстрокой
SUBSTRING	Возвращает для строки подстроку указанной длины с заданного символа
UNICODE	Возвращает Unicode-код левого символа строки
UPPER	Переводит все символы строки в верхний регистр

Использование функций для работы с числами

Краткий обзор математических функций

Название функции	**Действие, выполняемое функцией**
ABS	Вычисляет абсолютное значение числа
ACOS	Вычисляет арккосинус
ASIN	Вычисляет арксинус
ATAN	Вычисляет арктангенс
ATN2	Вычисляет арктангенс с учетом квадратов
CEILING	Выполняет округление вверх
COS	Вычисляет косинус угла
COT	Возвращает котангенс угла
DEGREES	Преобразует значение угла из радиан в градусы
EXP	Возвращает экспоненту
FLOOR	Выполняет округление вниз
LOG	Вычисляет натуральный логарифм
LOG10	Вычисляет десятичный логарифм
PI	Возвращает значение "пи"
POWER	Возводит число в степень
RADIANS	Преобразует значение угла из градуса в радианы
RAND	Возвращает случайное число
ROUND	Выполняет округление с заданной точностью
SIGN	Определяет знак числа
SIN	Вычисляет синус угла
SQUARE	Выполняет возведение числа в квадрат
SQRT	Извлекает квадратный корень
TAN	Возвращает тангенс угла

Использование функций для работы с типом дата/время

Краткий обзор основных функций для работы с датой и временем

Название функции	Действие, выполняемое функцией
DATEADD	Добавляет к дате указанное значение дней, месяцев, часов и т.д.
DATEDIFF	Возвращает разницу между указанными частями двух дат
DATENAME	Выделяет из даты указанную часть и возвращает ее в символьном формате
DATEPART	Выделяет из даты указанную часть и возвращает ее в числовом формате

DAY	Возвращает число из указанной даты
GETDATE	Возвращает текущее системное время
ISDATE	Проверяет правильность выражения на соответствие одному из возможных форматов ввода даты
MONTH	Возвращает значение месяца из указанной даты
YEAR	Возвращает значение года из указанной даты
MINUTE	Возвращает значение минут из указанной даты/времени
HOUR	Возвращает значение часов из указанной даты/времени
SECOND	Возвращает значение секунд из указанной даты/времени

Варианты заданий к лабораторной работе

Общие сведения

Для выполнения заданий ориентироваться на вариант и список номеров заданий.

Список вариантов заданий

Вариант	Список номеров упражнений												
1	1	6	11	16	21	26	31	36	41	46	51	56	61
2	2	7	12	17	22	27	32	37	42	47	52	57	62
3	3	8	13	18	23	28	33	38	43	48	53	58	63
4	4	9	14	19	24	29	34	39	44	49	54	59	64
5	5	10	15	20	25	30	35	40	45	50	55	60	64
6	6	11	16	21	26	31	36	41	46	51	56	61	1
7	7	12	17	22	27	32	37	42	47	52	57	62	2
8	8	13	18	23	28	33	38	43	48	53	58	63	3
9	9	14	19	24	29	34	39	44	49	54	59	64	4
10	10	15	20	25	30	35	40	45	50	55	60	65	5
11	2	6	12	16	22	26	32	36	42	46	52	56	62
12	1	5	11	15	21	25	31	35	41	45	51	55	61
13	3	7	13	17	23	27	33	37	43	47	53	57	63

Специальные знаки и простейшие операторы в Transact SQL

1. Проверить работу описанной установки SET QUOTED_IDENTIFIER.
2. Проверить работу описанной установки SET DATEFIRST.

Объявление переменных

3. Объявить переменную Perem1 типа денежный, а переменную Perem2 типа число с целой частью равной 8 и дробной частью равной 2.
4. Объявить переменную Perem1 типа строка длиной 100, а переменную Perem2 типа длинное целое.
5. Объявить переменную Perem1 типа динамическая строка с максимальной длиной 1000, а переменную Perem2 типа целое число.
6. Объявить переменную Perem1 типа строка длиной 30, а переменную Perem2 типа

число с целой частью равной 10 и дробной частью равной 3.

7. Объявить переменную Perem1 типа дата/ время, а переменную Perem2 типа число в диапазоне от 0 до 255.

Присвоение значений переменным и вывод значений на экран

8. Подсчитать среднюю цену закупленных книг (с помощью запроса SELECT) и умножить ее на значение 123,34, которое необходимо сохранить в отдельной переменной, вывести значение переменной на экран.

9. Подсчитать суммарную цену всех закупок книг, результат поместить в переменную, вывести значение переменной на экран.

10. Подсчитать количество книг в справочнике книг, результат поместить в переменную, вывести значение переменной на экран.

11. Определить минимальную дату рождения автора в справочнике авторов, результат поместить в переменную, вывести значение переменной на экран.

Сочетание ключевых слов SET и SELECT

12. Подсчитать количество поставщиков книг, результат поместить в переменную.

13. Подсчитать сумму закупок книг, результат поместить в переменную.

14. Подсчитать среднюю цену в таблице покупок книг, результат поместить в переменную.

15. Подсчитать максимальную стоимость книг в закупке, результат поместить в переменную.

Работа с датой и временем

16. Определить переменную Date1 типа дата/время. Присвоить ей значение даты 31.12.2016 в формате dd.mm.yyyy.

17. Определить переменную Date1 типа дата/время. Присвоить ей значение даты 31.12.2016 в формате mm.dd.yyyy.

18. Определить переменную Date1 типа дата/время. Присвоить ей значение даты 31.12.2016 в формате yyyy.mm.dd.

Создание временной таблицы через переменную типа TABLE

19. Создать локальную таблицу с названием TEMP и полями типа, дата/время, длинное целое, строка. Добавить в нее две записи с данными и вывести результат на экран.

20. Создать локальную таблицу с названием TEMP и полями типа длинное целое, строка и значением по умолчанию «введите что-нибудь», денежный. Добавить в нее две записи с данными и вывести результат на экран.

21. Создать локальную таблицу с названием TEMP и полями типа целое, динамическая строка, бит со значением по умолчанию «1». Добавить в нее две записи с данными и вывести результат на экран.

22. Создать локальную таблицу с названием TEMP и полями типа, дата/время, длинное целое, строка. Добавить в нее две записи с данными и вывести результат на экран.

23. Создать локальную таблицу с названием TEMP и полями типа, дата/время, длинное целое с автонаращиванием, динамическая строка. Добавить в нее две записи с данными и вывести результат на экран.

Преобразование типов переменных

24. Объявить переменные типа FLOAT, CHAR, TINYINT. Присвоить значения, соответствующие типам. Выполнить преобразование переменных типа FLOAT, CHAR, TINYINT в INT, DATETIME, BIT соответственно и вывести результат на экран.

25. Объявить переменные типа INT, DATETIME, BIT. Присвоить значения, соответствующие типам. Выполнить преобразование переменных типа INT, DATETIME, BIT в FLOAT, CHAR, TINYINT соответственно и вывести результат на экран.

26. Объявить переменные типа NUMERIC, VARCHAR, DATETIME. Присвоить значения, соответствующие типам. Выполнить преобразование переменных типа NUMERIC, VARCHAR, DATETIME в FLOAT, CHAR, BIGINT соответственно и вывести результат на экран.

27. Объявить переменные типа BIT, NVARCHAR, DATETIME. Присвоить значения, соответствующие типам. Выполнить преобразование переменных типа BIT, NVARCHAR, DATETIME в FLOAT, INT, BIGINT со ответственно и вывести результат на экран.

Условная конструкция IF

28. Подсчитать количество поставщиков в таблице Deliveries. Если их в таблице от 2 до 5, то ничего не сообщать, в противном случае вывести сообщение вида "В таблице ... поставщиков" (вместо многоточия поста вить точное количество поставщиков).

29. Подсчитать сумму закупок книг в таблице покупок. Если полученная сумма в диапазоне от 1000 до 5000, то ничего не сообщать, в против ном случае вывести сообщение вида "Сумма закупок = …" (вместо многоточия поставить точную сумму).

30. Подсчитать среднюю стоимость закупки книг в таблице покупок. Если полученная стоимость в диапазоне от 1000 до 5000, то ничего не сообщать, в противном случае вывести сообщение вида "Средняя стоимость закупки = …" (вместо многоточия поставить точную среднюю стоимость).

31. Определить минимальную стоимость закупки книг в таблице покупок. Если полученная стоимость в диапазоне от 200 до 300, то ничего не сообщать, в противном случае вывести сообщение вида "Минимальная стоимость закупки = …" (вместо многоточия поставить точную стоимость).

Цикл WHILE

32. Определить количество записей в таблице Authors. Пока записей меньше 15, делать в цикле добавление записи в таблицу с автоматическим наращиванием значения ключевого поля, а вместо имени автора ставить значение 'Автор не известен'.

33. Определить количество записей в таблице издательств. Пока записей меньше 20, делать в цикле добавление записи в таблицу с автоматическим наращиванием значения ключевого поля, а вместо названия издательства ставить значение 'не известно'.

34. Определить количество записей в таблице поставщиков. Пока записей меньше 17, делать в цикле добавление записи в таблицу с автоматическим наращиванием значения ключевого поля, а вместо названия поставщика ставить значение 'не известен'.

Объявление курсора

35. Создать статический курсор по данным таблицы Books с полями Code_book, Title_book.

36. Создать динамический курсор по данным таблицы поставщиков (таблица Deliveries) с полями Name_delivery, Name_company.

37. Создать статический курсор по данным таблицы Books и Authors с полями Code_book, Title_book, Name_author.

38. Создать статический курсор по данным таблицы Books и Publishing_house с полями Code_book, Title_book, Publish.

Операторы для работы с курсором

39. Создать динамический курсор для чтения по данным таблицы De liveries с полями Code_delivery, Name_delivery. Вывести данные 3-й записи.

40. Сделать текущей БД db_books. Поместить в курсор данные таблицы Purchases. Перебрать все записи таблицы Purchases. Просуммировать значения произведений полей Cost и Amount и результат сохранить в переменной Sum_table, которую после суммирования вывести на экран. За крыть и удалить из памяти курсор.

41. Объявить статический курсор по данным таблиц Authors и Books. Вывести данные 5-й записи.

Использование функций для работы со строковыми переменными

Для выполнения этого блока заданий в начале программы, которую вы создаете, объявите переменную типа varchar и присвойте ей в качестве значения строку с любым базовым текстом, который будет анализироваться и/или исправляться в заданиях.

42. Удалить в тексте лишние пробелы. Лишними считаются те, которые идут непосредственно за пробелом. Подсчитать количество исправлений.

43. Подсчитать количество встреч каждой из следующих букв: "а", "в", "и", "п" в базовом тексте.

44. Подсчитать доли процентов встречи следующих букв: "е", "о", если суммарный процент встречаемости всех этих букв равен 100% или процент встречаемости е% + о% равен 100%.

45. По правилам оформления машинописных текстов перед знаками .,!?:; пробелы не ставятся, но обязательно ставятся после этих знаков. Удалите лишние пробелы. Подсчитать количество исправлений.

46. По правилам оформления машинописных текстов перед знаками .,!?:; пробелы не ставятся, но обязательно ставятся после этих знаков. Расставьте недостающие пробелы. Подсчитать количество исправлений.

47. Найти из исходного текста второе предложение и вернуть его в переменную Perem, а также вывести на экран весь исходный текст и найденное предложение.

48. Удалить из базового текста 2, 4, 6, 8 слова.

49. Удалить из базового текста 3, 5, 7, 10 слова.

50. Вставить в базовый текст вместо букв «а» «АА».

51. Вставить в базовый текст вместо букв «е» и «о» «ББ».

52. Поменять местами первое и последнее слова в базовом тексте.

Использование функций для работы с числами

53. Вывести значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$v = v_0 \cdot e^{\sqrt{\frac{R \cdot T}{45}}}$$

.

54. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = 2^x \cdot \exp(\ln(x^2))$$

55. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = \frac{sin(a)}{x^2 - b^3} \cdot a.$$

56. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = \sum_{n=1}^{10} I_n \cdot a$$

.

57. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = \frac{tg(a)}{a + b - c} \cdot \sqrt{a \cdot b \cdot c}.$$

58. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = \sqrt{\sin(a) \cdot \exp(b \cdot c)}$$

59. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = x^4 \cdot ln(a) - b \cdot c$$

60. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = \frac{\sqrt{x - a}}{b^3}$$

61. Подсчитать значение формулы, переменные которой нужно описать и присвоить произвольные значения.

$$y = \frac{a \cdot cos(x)}{b^2 - a^2} \cdot sin(x)$$

Использование функций для работы с типом дата/время

62. Вывести на экран название текущего месяца и текущее время. За писать в таблицу Purchases в поле Date_order одинаковую дату поступления, которая равна 12.03.2000.

63. Разобрать на отдельные составляющие текущую дату и время и вывести значения на экран в следующем порядке (вместо многоточий): "Сегодня: День = …, Месяц = …, Год = …, Часов = …, Минут = …,Секунд= …"

64. В исходный текст, сохраненный в переменной Perem, после слова "время" вставить текущее время. Результат сохранить в той же переменной Perem и вывести на экран.

Контрольные вопросы

1. Как записываются комментарии в языке Transact SQL?

2. Какие типы встроенных функций существуют в TRANSACT SQL? К какому типу относятся функции RTRIM, STR и UPPER? К какому типу относятся функции GETDATE, MONTH и DATEADD?
3. В чем отличие присвоения значений переменным с помощью SET и SELECT?
4. Чем отличаются структуры Table и Cursor?
5. Опишите основные приемы работы с набором строк Cursor.
6. Какие вы знаете операторы условных конструкций и циклов в Transact SQL?

Лабораторная работа №9(5)

СОЗДАНИЕ ХРАНИМЫХ ПРОЦЕДУР В MICROSOFT SQL SERVER

Цель работы – научиться создавать и использовать хранимые процедуры на сервере БД.

Содержание работы:

1. Проработка всех примеров, анализ результатов их выполнения в утилите SQL Server Management Studio. Проверка наличия созданных процедур в текущей БД.
2. Выполнение всех примеров и заданий по ходу лабораторной работы.
3. Выполнение индивидуальных заданий по вариантам.

Пояснения к выполнению работы

Для освоения программирования хранимых процедур используем пример базы данных с названием **DB_Books**, которая была создана в лабораторной работе №7(3).

Хранимые процедуры представляют собой набор команд, состоящий из одного или нескольких операторов SQL или функций и сохраняемый в базе данных в откомпилированном виде.

Типы хранимых процедур

Системные хранимые процедуры предназначены для выполнения различных административных действий. Практически все действия по администрированию сервера выполняются с их помощью. Можно сказать, что системные хранимые процедуры являются интерфейсом, обеспечивающим работу с системными таблицами. Системные хранимые процедуры имеют префикс sp_, хранятся в системной базе данных и могут быть вызваны в контексте любой другой базы данных.

Пользовательские *хранимые процедуры* реализуют те или иные действия. *Хранимые процедуры* – полноценный объект базы данных. Вследствие этого каждая *хранимая процедура* располагается в конкретной базе данных, где и выполняется.

Временные *хранимые процедуры* существуют лишь некоторое время, после чего автоматически уничтожаются сервером. Они делятся на локальные и глобальные. Локальные временные *хранимые процедуры* могут быть вызваны только из того соединения, в котором созданы. При создании такой процедуры ей необходимо дать имя, начинающееся с одного символа #. Как и все временные объекты, *хранимые процедуры* этого типа автоматически удаляются при отключении пользователя, перезапуске или остановке сервера. Глобальные временные *хранимые процедуры* доступны для любых соединений сервера, на котором имеется такая же процедура. Для ее определения достаточно дать ей имя, начинающееся с символов ##. Удаляются эти процедуры при перезапуске или остановке сервера, а также при закрытии соединения, в контексте которого они были созданы.

Создание, изменение хранимых процедур

При создании хранимой процедуры следует учитывать, что она будет иметь те же

права доступа к объектам базы данных, что и создавший ее пользователь; определение параметров хранимой процедуры, хранимые процедуры могут обладать входными и выходными параметрами; разработка кода хранимой процедуры. Код процедуры может содержать последовательность любых команд SQL, включая вызов других хранимых процедур.

Синтаксис оператора создания новой или изменения имеющейся хранимой процедуры в обозначениях MS SQL Server:

```
{CREATE | ALTER PROC[EDURE] имя_процедуры [;номер] [{@имя_параметра тип_данных } [VARYING ] [=default][OUTPUT] ][,...n] [WITH { RECOMPILE | ENCRYPTION }] [FOR REPLICATION] AS sql_оператор [...n]
```

Рассмотрим параметры данной команды.

Используя префиксы sp, #, ##, создаваемую процедуру можно определить в качестве системной или временной. Как видно из синтаксиса команды, не допускается указывать имя владельца, которому будет принадлежать создаваемая процедура, а также имя базы данных, где она должна быть размещена. Таким образом, чтобы разместить создаваемую хранимую процедуру в конкретной базе данных, необходимо выполнить команду CREATE PROCEDURE в контексте этой базы данных. При обращении из тела хранимой процедуры к объектам той же базы данных можно использовать укороченные имена, т. е. без указания имени базы данных. Когда же требуется обратиться к объектам, расположенным в других базах данных, указание имени базы данных обязательно.

Для передачи входных и выходных данных в создаваемой хранимой процедуре имена параметров должны начинаться с символа @. В одной хранимой процедуре можно задать множество параметров, разделенных запятыми. В теле процедуры не должны применяться локальные переменные, чьи имена совпадают с именами параметров этой процедуры.

Для определения типа данных параметров хранимой процедуры подходят любые типы данных SQL, включая определенные пользователем. Однако тип данных CURSOR может быть использован только как выходной параметр хранимой процедуры, т.е. с указанием ключевого слова OUTPUT.

Наличие ключевого слова OUTPUT означает, что соответствующий параметр предназначен для возвращения данных из хранимой процедуры. Однако это вовсе не означает, что параметр не подходит для передачи значений в хранимую процедуру. Указание ключевого слова OUTPUT предписывает серверу при выходе из хранимой процедуры присвоить текущее значение параметра локальной переменной, которая была указана при вызове процедуры в качестве значения параметра. Отметим, что при указании ключевого слова OUTPUT значение соответствующего параметра при вызове процедуры может быть задано только с помощью локальной переменной. Не разрешается использование любых выражений или констант, допустимое для обычных параметров.

Ключевое слово VARYING применяется совместно с параметром OUTPUT, имеющим тип CURSOR. Оно определяет, что выходным параметром будет результирующее множество.

Ключевое слово DEFAULT представляет собой значение, которое будет принимать соответствующий параметр по умолчанию. Таким образом, при вызове процедуры можно не указывать явно значение соответствующего параметра.

Так как сервер кэширует план исполнения запроса и компилированный код, при последующем вызове процедуры будут использоваться уже готовые значения. Однако в некоторых случаях все же требуется выполнять перекомпиляцию кода процедуры. Указание ключевого слова RECOMPILE предписывает системе создавать план выполнения хранимой процедуры при каждом ее вызове.

Параметр FOR REPLICATION востребован при репликации данных и включении создаваемой хранимой процедуры в качестве статьи в публикацию.

Ключевое слово ENCRYPTION предписывает серверу выполнить шифрование кода хранимой процедуры, что может обеспечить защиту от использования авторских алгоритмов, реализующих работу хранимой процедуры.

Ключевое слово AS размещается в начале собственно тела хранимой процедуры. В теле процедуры могут применяться практически все команды SQL, объявляться транзакции, устанавливаться блокировки и вызываться другие хранимые процедуры. Выход из хранимой процедуры можно осуществить посредством команды RETURN.

Удаление хранимой процедуры

DROP PROCEDURE {имя_процедуры} [,...n]

Выполнение хранимой процедуры

Для выполнения хранимой процедуры используется команда:

```
[[ EXEC[UTE] имя_процедуры [;номер] [[@имя_параметра=]{значение
| @имя_переменной} [OUTPUT ]|[DEFAULT ]][,...n]
```

Если вызов хранимой процедуры не является единственной командой в пакете, то присутствие команды EXECUTE обязательно. Более того, эта команда требуется для вызова процедуры из тела другой процедуры или триггера.

Использование ключевого слова OUTPUT при вызове процедуры разрешается только для параметров, которые были объявлены при создании процедуры с ключевым словом OUTPUT.

Когда же при вызове процедуры для параметра указывается ключевое слово DEFAULT, то будет использовано значение по умолчанию. Естественно, указанное слово DEFAULT разрешается только для тех параметров, для которых определено значение по умолчанию.

Из синтаксиса команды EXECUTE видно, что имена параметров могут быть опущены при вызове процедуры. Однако в этом случае пользователь должен указывать значения для параметров в том же порядке, в каком они перечислялись при создании процедуры. Присвоить параметру значение по умолчанию, просто пропустив его при перечислении, нельзя. Если же требуется опустить параметры, для которых определено значение по умолчанию, достаточно явного указания имен параметров при вызове хранимой процедуры. Более того, таким способом можно перечислять параметры и их значения в произвольном порядке.

Отметим, что при вызове процедуры указываются либо имена параметров со значениями, либо только значения без имени параметра. Их комбинирование не допускается.

Использование RETURN в хранимой процедуре

Позволяет выйти из процедуры в любой точке по указанному условию, а также позволяет передать результат выполнения процедуры числом, по которому можно судить о качестве и правильности выполнения процедуры. Пример создания процедуры без параметров:

```
CREATE PROCEDURE Count_Books AS
Select count(Code_book) from Books
Go
```

Задание 1. Создайте данную процедуру в разделе Stored Procedures базы данных DB_Books через утилиту SQL server Management Studio. Запустите ее с помощью команды

```
EXEC Count_Books
```

Проверьте результат.

Пример создания процедуры с входным параметром:

```
CREATE PROCEDURE Count_Books_Pages @Count_pages as Int AS
Select count(Code_book) from Books WHERE Pages>=@Count_pages
Go
```

Задание 2. Создайте данную процедуру в разделе Stored Procedures базы данных DB_Books через утилиту SQL server Management Studio. Запустите ее с помощью команды

```
EXEC Count_Books_Pages 100
```

Проверьте результат.

Пример создания процедуры с входными параметрами:

```
CREATE PROCEDURE Count_Books_Title @Count_pages as Int, @Title AS
Char(10)
AS
Select count(Code_book) from Books WHERE Pages>=@Count_pages AND
Title_book LIKE @Title
Go
```

Задание 3. Создайте данную процедуру в разделе Stored Procedures базы данных DB_Books через утилиту SQL server Management Studio. Запустите ее с помощью команды

```
EXEC Count_Books_Title 100, 'П%'
```

Проверьте результат.

Пример создания процедуры с входными параметрами и выходным параметром:

```
CREATE PROCEDURE Count_Books_Itogo @Count_pages Int, @Title
Char(10), @Itogo Int OUTPUT
AS
Select @Itogo = count(Code_book) from Books
WHERE Pages>=@Count_pages AND Title_book LIKE @Title
Go
```

Задание 4. Создайте данную процедуру в разделе Stored Procedures базы данных DB_Books через утилиту SQL server Management Studio. Запустите с помощью набора команд:

```
Declare @q As int
EXEC Count_Books_Itogo 100, 'П%', @q output select @q
```

Проверьте результат.

Пример создания процедуры с входными параметрами и RETURN:

```
CREATE PROCEDURE checkname @param int
AS
IF (SELECT Name_author FROM authors WHERE Code_author = @param) =
'Пушкин А.С.'
RETURN 1
ELSE
RETURN 2
```

Задание 5. Создайте данную процедуру в разделе Stored Procedures базы данных DB_Books через утилиту SQL server Management Studio. Запустите ее с помощью команд:

```
DECLARE @return_status int
EXEC @return_status = checkname 1 SELECT 'Return Status' =
@return_status
```

Пример создания процедуры без параметров для увеличения значения ключевого поля в таблице Purchases в 2 раза:

```
CREATE PROC update_proc
AS
UPDATE Purchases SET Code_purchase = Code_purchase*2
```

Процедура не возвращает никаких данных.

Задание 6. Создайте данную процедуру в разделе Stored Procedures базы данных DB_Books через утилиту SQL server Management Studio. Запустите ее с помощью команды

```
EXEC update_proc
```

Пример процедуры с входным параметром для получения всей информации о конкретном авторе:

```
CREATE PROC select_author @k CHAR(30)
AS
SELECT * FROM Authors WHERE name_author=@k
```

Задание 7. Создайте данную процедуру в разделе Stored Procedures базы данных DB_Books через утилиту SQL server Management Studio. Запустите ее с помощью команд:

```
EXEC select_author 'Пушкин А.С.'
```

или

```
select_author @k='Пушкин А.С.'
```

или

```
EXEC select_author @k='Пушкин А.С.'
```

Пример создания процедуры с входным параметром и значением по умолчанию для увеличения значения ключевого поля в таблице Purchases в заданное количество раза (по умолчанию в 2 раза):

```
CREATE PROC update_proc @p INT = 2
AS
UPDATE Purchases SET Code_purchase = Code_purchase *@p
```

Процедура не возвращает никаких данных.

Задание 8. Создайте данную процедуру в разделе Stored Procedures базы данных DB_Books через утилиту SQL server Management Studio. Запустите ее с помощью команд:

```
EXEC update_proc 4 или
EXEC update_proc @p = 4 или
EXEC update_proc --будет использовано значение по умолчанию.
```

Пример создания процедуры с входным и выходным параметрами. Создать процедуру для определения количества заказов, совершенных за указанный период:

```
CREATE PROC count_purchases
@d1 SMALLDATETIME, @d2 SMALLDATETIME,
@c INT OUTPUT AS
SELECT @c=count(Code_purchase) from Purchases
WHERE Date_order BETWEEN @d1 AND @d2
SET @c = ISNULL(@c,0)
```

Задание 9. Создайте данную процедуру в разделе Stored Procedures базы данных DB_Books через утилиту SQL server Management Studio. Запустите ее с помощью команд:

```
DECLARE @c2 INT
EXEC count_purchases '01-jun-2006', '01-jul-2006', @c2
OUTPUT SELECT @c2
```

Варианты заданий к лабораторной работе

Общие положения

В утилите SQL Server Management Studio создать новую страницу для кода (кнопка «Создать запрос»). Программно сделать активной созданную БД DB_Books с помощью оператора Use. Создать хранимые процедуры с помощью операторов Create procedure, причем самостоятельно определить имена процедур. Каждая процедура будет выполнять по одному SQL запросу. Причем код SQL запросов нужно изменить таким образом, чтобы в них можно было передавать значения полей, по которым осуществляется поиск.

Например, исходное задание и запрос:

/*Выбрать из справочника поставщиков (таблица Deliveries) названия компаний, телефоны и ИНН (поля Name_company, Phone и INN), у которых название компании (поле Name_company) 'ОАО МИР'.

SELECT Name_company, Phone, INN FROM Deliveries WHERE

Name_company = 'ОАО МИР'

*/

--В данной работе будет создана процедура:

```
CREATE PROC select_name_company @comp CHAR(30) AS
```

```
SELECT Name_company, Phone, INN FROM Deliveries WHERE Name_company
= @comp
```

--Для запуска процедуры используется команда: EXEC select_name_company 'ОАО МИР'

Список заданий

Задания выполнять для индивидуальной БД, созданной в лабораторной работе №7(3).

Вариант 1

1. Вывести список сотрудников, у которых есть хотя бы один ребенок.
2. Вывести список детей, которым выдали подарки в указанный период.
3. Вывести список родителей, у которых есть дети возрастом до 14 лет.
4. Вывести информацию о подарках со стоимостью больше указанного числа, отсортированных по дате.

Вариант 2

1. Вывести список приборов с указанным типом.
2. Вывести количество отремонтированных приборов и общую стоимость ремонтов у указанного мастера.
3. Вывести список владельцев приборов и количество их обращений, отсортированный по количеству обращений по убыванию.
4. Вывести информацию о мастерах с разрядом больше указанного числа или с датой приема на работу меньше указанной даты.

Вариант 3

1. Вывести список цветков с указанным типом листа.
2. Вывести список кодов продаж, по которым продано цветов на сумму больше указанного числа.
3. Вывести дату продажи, сумму, продавца и цветок по указанному коду продажи.
4. Вывести список цветов и сорт для цветов с высотой больше указанного числа или цветущий.

Вариант 4

1. Вывести список лекарств с указанным показанием к применению.
2. Вывести список дат поставок, по которым продано больше указанного числа одноименного лекарства.
3. Вывести дату поставки, сумму, ФИО руководителя от поставщика и название лекарства по коду поступления больше указанного числа.
4. Вывести список лекарств и единицы измерения для лекарств с количеством в упаковке больше указанного числа или кодом лекарства меньше определенного значения.

Вариант 5

1. Вывести список сотрудников с указанной должностью.
2. Вывести список списанного оборудования по указанной причине.
3. Вывести дату поступления, название оборудования, ФИО ответственного и дату списания для оборудования, списанного в указанный период.
4. Вывести список оборудования с указанным типом или с датой поступления больше определенного значения.

Вариант 6

1. Вывести список блюд с весом больше указанного числа.
2. Вывести список продуктов, в названии которых встречается указанный фрагмент слова.
3. Вывести объем продукта, название блюда, название продукта с кодом блюда от указанного начального значения по определенному конечному значению.
4. Вывести порядок приготовления блюда и название блюда с количеством углеводов больше определенного значения или количеством калорий больше указанного значения.

Вариант 7

1. Вывести список сотрудников с указанной должностью.
2. Вывести список документов, в содержании которых встречается указанный фрагмент слова.
3. Вывести дату регистрации, тип документа, ФИО регистратора и название организации для документов, зарегистрированных в указанный период.
4. Вывести список зарегистрированных документов с определенным типом документа или с датой регистрации больше указанного значения.

Вариант 8

1. Вывести список сотрудников с указанной причиной увольнения.
2. Вывести список документов с датой регистрации в указанный период.
3. Вывести дату регистрации, причину увольнения, ФИО сотрудника для документов, зарегистрированных в указанный период.
4. Вывести список зарегистрированных документов с кодом документа в указанном диапазоне.

Вариант 9

1. Вывести список сотрудников, бравших отпуск указанного типа.
2. Вывести список документов с датой регистрации в указанный период.
3. Вывести дату регистрации, тип отпуска, ФИО сотрудника для документов, зарегистрированных в указанный период.
4. Вывести список зарегистрированных документов с кодом документа в указанном диапазоне.

Вариант 10

1. Вывести список сотрудников с указанной должностью.
2. Вывести список документов, в содержании которых встречается указанный фрагмент слова.
3. Вывести дату регистрации, тип документа, ФИО отправителя и название организации для документов, зарегистрированных в указанный период.
4. Вывести список зарегистрированных документов с указанным типом документа или с кодом документа меньше определенного значения.

Вариант 11

1. Вывести список сотрудников, назначенных на указанную должность.
2. Вывести список документов с датой регистрации в указанный период.
3. Вывести дату регистрации, должность, ФИО сотрудника для документов, зарегистрированных в указанный период.
4. Вывести список зарегистрированных документов с кодом документа в указанном диапазоне.

Вариант 12

1. Вывести список оборудования с указанным типом.
2. Вывести список оборудования, которое брал в прокат определенный клиент.
3. Вывести список лиц, бравших оборудование в прокат и количество их обращений, отсортированный по количеству обращений по убыванию.
4. Вывести информацию о клиентах, отсортированных по адресам.

Вариант 13

1. Вывести список поставщиков, отсортированный по количеству закупленного у них оборудования.
2. Вывести список закупленного оборудования указанного типа.
3. Вывести дату покупки, название оборудования, цену покупки, данные поставщика для оборудования, закупленного в указанный период.
4. Вывести список оборудования с указанным типом или с датой поступления больше определенного значения.

Контрольные вопросы

1. Что такое хранимая процедура?
2. Где выполняются хранимые процедуры?
3. Как активизируются хранимые процедуры?
4. В чем преимущества использования хранимых процедур?
5. Какие типы хранимых процедур имеются в SQL Server?
6. Что такое триггер?
7. В чем преимущества использования триггеров?

Лабораторная работа 10(6)
СОЗДАНИЕ КЛИЕНТСКОЙ ЧАСТИ ПРИЛОЖЕНИЯ ДЛЯ ПРОСМОТРА, РЕДАКТИРОВАНИЯ ДАННЫХ БД. ВЫЗОВ ХРАНИМЫХ ПРОЦЕДУР ИЗ КЛИЕНТСКОЙ ЧАСТИ СОЗДАНИЕ ОТЧЕТНЫХ ФОРМ В КЛИЕНТСКОМ ПРИЛОЖЕНИИ

Цель работы **–** научиться создавать клиентское приложение для работы с базой данных с применением встроенных инструментов на Visual C#, научиться создавать формы отчетных документов по данным БД

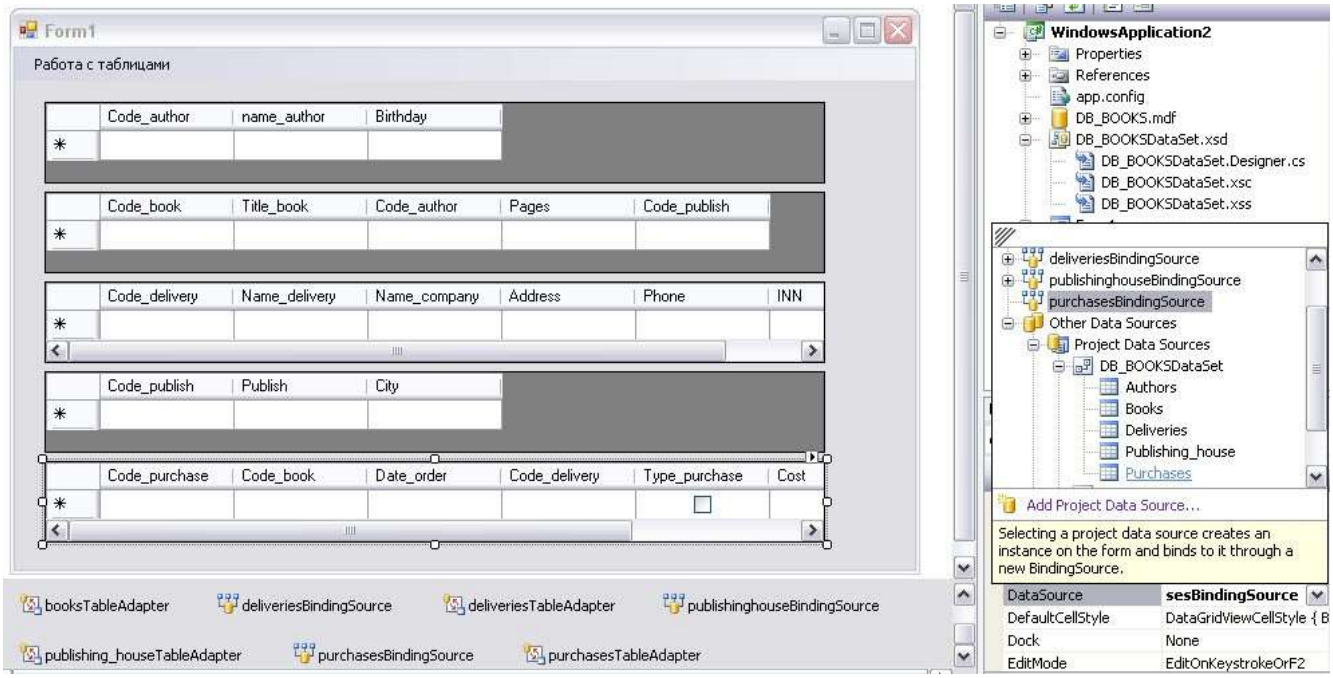
Содержание работы:

1. Выполнение всех заданий по ходу лабораторной работы.
2. Выполнение индивидуальных заданий.

Пояснения к выполнению работы

Для создания клиентского приложения на Visual C# (примеры приводятся для версии Visual C# не менее 2005) используем пример базы данных с названием **DB_Books**.

1. В проекте выбираем меню Tools => Connect to DataBase.
2. В открывшемся окне в поле Data Source ставим Microsoft SQL Server, в поле Server Name – SQLEXPRESS, далее в поле Select or enter DB name выбирете имя БД, к которой будем подключаться, и нажмите OK.
3. Теперь открыв окно Server explorer можно увидеть подключенную БД. Нажав на нее, в окне свойств копируем Connection String, она еще пригодится.
4. На форму добавить 5 компонентов типа DataGridView (переименовать компоненты на Purchases, Books, Authors, Deliveries, Publish).
5. Во вкладке Data выберем Add New Data Source. В появившемся окне выберем DataBase и нажмем Next. Выбераем нашу БД, жмем Next. В появившемся окне поставим галочку на пункте Table (выбераем все созданные таблици). Жмем Finish.
5. У каждой таблице DataGridView изменим свойство DataSource на соответствующие названию этой таблицы:



6. На основной форме (Form1) добавить компонент. В редакторе меню сделать первый пункт «Работа с таблицами» и в подменю пункты: «Авторы», «Книги», «Издательства», «Поставщики», «Поставки».

7. Создать пять форм, каждую из которых назвать: FormAuthors, FormPurchases, FormBooks, FormDeliveries, FormPublish.

8. На основной форме в подпунктах меню в соответствующих методах Click вызвать соответствующие формы с помощью кода: для FormAuthors:

```
FormAuthors myForm2 = new FormAuthors();
myForm2.Show();
```

для FormPurchases:

```
FormPurchases myForm3 = new FormPurchases();
myForm3.Show();
```

для FormBooks:

```
FormBooks myForm4 = new FormBooks();
myForm4.Show();
```

для FormDeliveries:

```
FormDeliveries myForm5 = new
FormDeliveries(); myForm5.Show();
```

для FormPublish:

```
FormPublish myForm6 = new FormPublish();
myForm6.Show();
```

9. На формы FormAuthors, FormPurchases, FormBooks, FormDeliveries, FormPublish добавить по паре компонент типа DataGridView и BindingNavigator. Настроить у DataGridView свойство DataSource для связи с соответствующим источником данных. Затем необходимо настроить у BindingNavigator свойство BindingSource для связи с созданной таблицей(значение должно совпадать со значением свойства элемента DataGridView).

10. Проверить работу приложения.

11. На форму FormBooks добавить 3 компонента типа TextBox и 2 компонента **ComboBox**.

У 1-го компонента **TextBox** изменить свойства:

(DataBinding) Text booksBindingSource Code_book

У 2-го компонента **TextBox** изменить свойства:

(DataBinding) Text booksBindingSource Title_book

У 1-го компонента **ComboBox** изменить свойства:

(DataBinding) SelectedValue booksBindingSource – Code_author
DataSource authorsBindingSource DisplayMember name_author
ValueMember Code_author

У 3-го компонента **TextBox** изменить свойства:

(DataBinding) Text booksBindingSource Pages

У 2-го компонента **ComboBox** изменить свойства:

(DataBinding) SelectedValue booksBindingSource – Code_publish
DataSource publishinghouseBindingSource
DisplayMember Publish
ValueMember Code_publish

12. У компонента DataGridView убрать все галочки со свойств редактирования и добавления.

13. На форму FormBooks добавить компонент типа Button (кнопка обновления данных), свойство **Text** изменить на «Обновить» и прописать событие Click:

```
this.Validate();
this.booksBindingSource.EndEdit();
this.booksTableAdapter.Update(this.dB_BOOKSDataSet.Books);
```

14. Аналогично для остальных форм добавить элементы типа TextBox

15. Проверить работу приложения.

16. На форму FormBooks добавить 5 компонентов типа Button.

У 1-го компонента Button изменить свойства и метод:

Text Фильтр по текущему издательству;

В методе **Click** кнопки написать код:

```
int bb = dataGridView1.CurrentCell.RowIndex;
booksBindingSource.Filter = "Code_Publish = " +
dataGridView1[4,bb].Value;.
```

У 2-го компонента Button изменить свойства и метод:

Text Фильтр по текущему названию книги.

В методе **Click** кнопки написать код:

```
int bb = dataGridView1.CurrentCell.RowIndex;
booksBindingSource.Filter = "Title_book = " +
dataGridView1[1, bb].Value;.
```

У 3-го компонента Button изменить свойства и метод:

Text Фильтр по текущему автору.

В методе **Click** кнопки написать код:

```
int bb = dataGridView1.CurrentCell.RowIndex;
booksBindingSource.Filter = "Code_Author = " +
dataGridView1[0, bb].Value;.
```

У 4-го компонента Button изменить свойства и метод:

Text Фильтр по количеству книг.

В методе **Click** кнопки написать код:

```
int bb = dataGridView1.CurrentCell.RowIndex;
booksBindingSource.Filter = "Pages = " +
dataGridView1[3, bb].Value;.
```

У 5-го компонента Button изменить свойства и метод:

Text Снять фильтр.

В методе **Click** кнопки написать код:

```
booksBindingSource.Filter = "";.
```

17. Аналогично для остальных форм добавить элементы типа Button, которые будут запускать фильтры по соответствующим значениям полей текущей записи в объекте Grid.

18. Проверить работу приложения.

19. Создать форму, назвать FormProcedure.

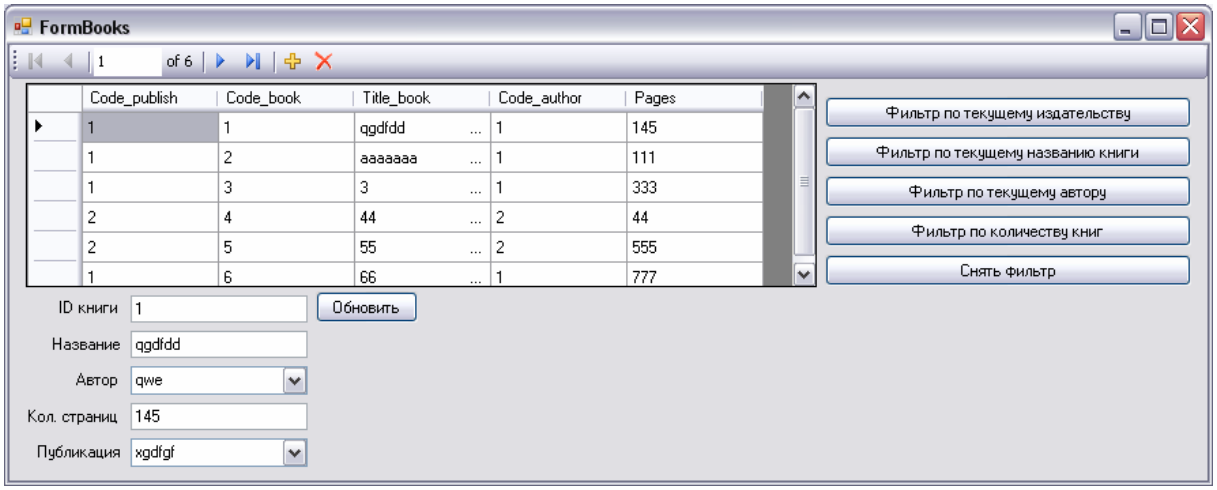


Рис. 1. Пример расположения компонентов на форме FormBooks

20. Добавить на главной форме в меню пункт с названием **Работа с процедурами**. В методе Click пункта меню написать код для запуска формы **FormProcedure**.

21. Зайти Tool -> Choose Toolbox Items. Поставить галочки на элементах SqlCommand и SqlConnection, применить изменения.

22. Добавить на форму компонент SqlConnection и в свойстве ConnectionString выбрать DB_DOOK.mdf

23. Теперь можно подключить хранимую процедуру Count_purchases. На форму **FormProcedure** добавить компонент SqlCommand. Изменить следующие его свойства:

Connection на SqlConnection1;

CommandType на StoredProcedure;

CommantText на Count_purchases.

24. У компонента SqlCommand1 выбрать свойтсво Parameters и в свойствах каждого входного параметра исправить свойство SqlDbType – на DateTime, а для выходного параметра свойство Value – Int. Также, если параметр со значением ReturnValue (параметр Direction) не создан, то необходимо создать его (он должен быть на самом верху) и задать ему имя @ReturnValue со свойством SqlDbType Int.

25. На форму **FormProcedure** добавить 3 компонента типа TextBox (имена соответственно TextBox1, TextBox2, TextBox3) и 1 компонент типа Button. Рядом с каждым компонентом TextBox поставить Label и исправить их свойства Text соответственно на «Количество покупок за указанный период», «Введите дату начала периода», «Введите дату конца периода».

26. На кнопке поменять название на «Выполнить запрос». В методе Click кнопки написать следующий код:

```
int count_save;
sqlCommand1.Parameters["@d1"].Value =
Convert.ToDateTime(textBox1.Text);
sqlCommand1.Parameters["@d2"].Value =
Convert.ToDateTime(textBox2.Text);
sqlConnection1.Open();
sqlCommand1.ExecuteNonQuery();
sqlConnection1.Close();
count_save = (int)sqlCommand1.Parameters["@ReturValue"].Value;
textBox3.Text = Convert.ToString(count_save);
```

27. Проверить работу приложения.

Отчеты во многом похожи на формы и тоже позволяют получить результаты работы запросов в наглядной форме, но только не на экране, а в виде распечатки на принтере. Таким образом, в результате работы отчета создается *бумажный документ*.

На Visual C# есть несколько способов создания отчетов. Один из способов создание отчетов это использование генератора отчета FastReport. Генератор можно скачать с официального сайта компании https://www.fast-report.com/ru/download/. После установки генератора необходимо перезапустить Visual C#. Затем необходимо добавить компоненты FastReport.

Для создания отчета необходимо поместить компонент Report на главную форму. После этого двойным щелчком нажать на компонент и выбрать данные, которые нам нужны для составления отчета. После этого откроется сам редактор отчетов. Для сохранения изменений нужно просто сохранить файл отчета в любом месте.

Структура отчета

Отчеты состоят из разделов или секций (Bands), а разделы могут со-

держать элементы управления. Для настройки разделов надо нажать на рабочей области на кнопку «Настроить бэнды».

1. Структура отчета состоит из следующих разделов: *заголовок отчета, подвал отчета, заголовок страницы, подвал страницы, область данных, заголовок колонки, подвал колонки, фоновый.*

2. Раздел ***заголовок отчета*** служит для печати общего заголовка отчета.

3. Раздел ***заголовок страницы*** можно использовать для печати подзаголовков, если отчет имеет сложную структуру и занимает много страниц.

Здесь можно также помещать и номера страниц, если это не сделано в нижнем колонтитуле.

4. В ***области данных*** размещают элементы управления, связанные с содержимым полей таблиц базы. В эти элементы управления выдаются данные из таблиц для печати на принтере. Эти разделы будут на печати воспроизводиться столько раз, сколько записей присутствует в привязанном запросе или таблице.

5. Раздел ***подвал страницы*** используют для тех же целей, что и раздел заголовок страницы. Можно использовать для подстановки полей для подписей должностных лиц, если есть необходимость подписывать отчет на каждой странице.

6. Разделы ***колонок*** используют для размещения дополнительной информации или итоговой информации по всем данным отчета. Печатается сверху или снизу области данных.

7. Для предварительного просмотра отчета в том виде, как он будет расположен на бумаге, необходимо вызвать метод **Show** компонента **Report (**на главной форме в меню добавить раздел и в методе **Click** написать этот метод, например, **Report1.Show())**. Пример отчета в режиме «Конструктор» представлен на рис 2, а в режиме предварительного просмотра – на рис. 3.

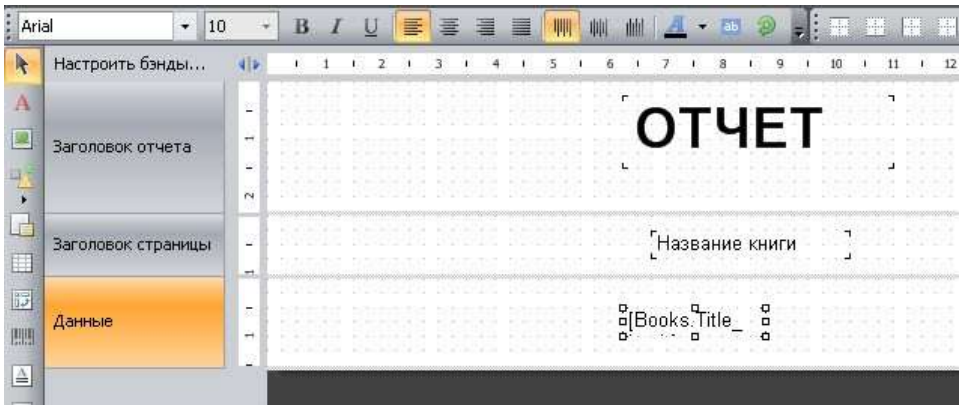


Рис 2. Пример отчета в режиме «Конструктор»

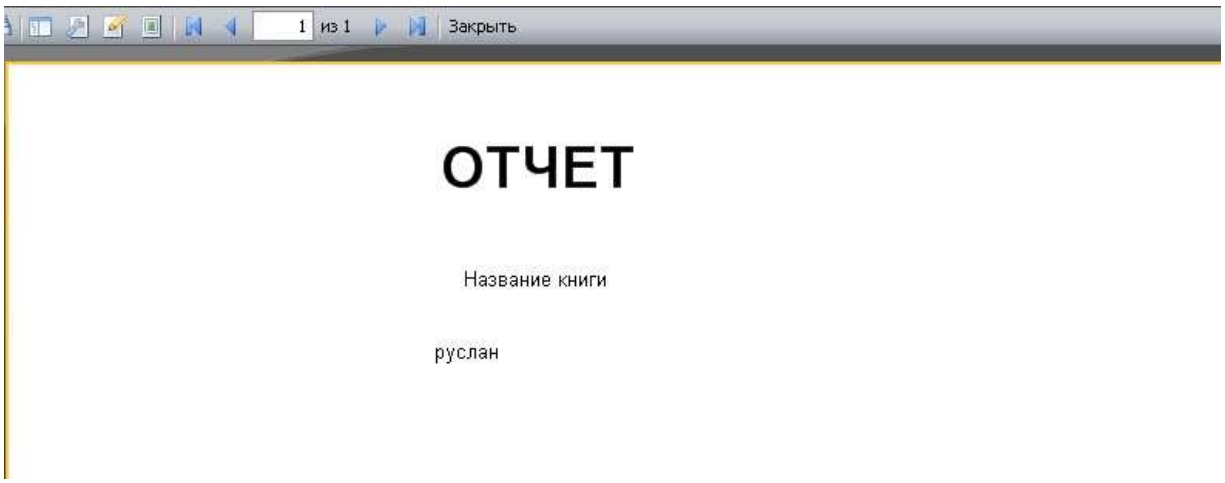


Рис. 3. Пример отчета в предварительном просмотре

Задание 1. Создание отчета в табличной форме, который выбирает из таблицы Books все поля, кроме кодов, из таблицы Publish_house − название издательства и место издательства, из таблицы Authors − имя автора.

1. В проекте на главной форме в меню добавить пункт меню **Отчеты**, а также подпункты:

Отчет в табличной форме;

Отчет в свободной форме;

Отчет с группировкой по двум таблицам.

2. В проекте на главную форму добавить 3 компонента **Report**

3. У первого компонента **Report** изменить **DataSet** (стрелка на компоненте Task -> Select DataSet) на соответствующие данные необходимые для отчета. Открыть окно дизайна отчета (двойным щелчком по компонен-
ту).

4. В свойствах включить такие разделы (настроить бэнды), как ***заголовок отчета, заголовок данных, данные***.

5. В разделе **заголовок отчета** разместить метку (компонент **Текст**). В свойствах изменить его внешний вид и подпись «Пример табличного отчета».

6. В разделе **заголовок данных** установить компонент **Таблица** (для имитации обрамления шапки таблицы) и написать в ней **Название книги, Автор, Издательство**.

7. На раздел **данные** перетянуть объекты с панели *данные* по следующему пути:

Источники данных -> Books -> title_book

Источники данных -> Books -> Autors -> name_autor

Источники данных -> Books -> Publishing_house -> publish

Расположить компоненты симметрично под надписями в таблице.

8. В главной форме приложения в подпункте **Отчет в табличной**

форме в методе Click написать команду: **Report1.Show().**

9. Запустить приложение, проверить работу.

Задание 2. Создание отчета в свободной форме с данными из первого задания. Создадим карточку книги для библиотечной картотеки.

Особенность отчета в свободной форме в том, что он создает шаблон на каждую отдельную запись таблицы, другими словами, он создается по документам, у которых нет шапки и примечаний. Примером таких документов может служить приходный или расходный кассовый ордер, этикетка для товара или ценник в магазине, пригласительное письмо и т.д.

1. У второго компонента **Report** установить свойства **DataSet** на необходимые. В свойствах (настроить бэнды) включить раздел ***Данные***.
2. На раздел **данные** перетянуть объекты с панели *данные* по следующему пути:

Источники данных -> Books -> title_book

Источники данных -> Autors -> name_autor

Источники данных -> Publishing_house -> publish

3. В главной форме приложения в подпункте **Отчет в свободной форме** в методе Click написать команду: **Report2.Show().**
4. Запустить приложение, проверить работу.

Задание 3. Создание отчета по двум таблицам. Создадим отчет с группировкой, в котором сначала будут выводиться данные автора книги из таблицы Authors, а затем список книг, которые написал этот автор.

1. У третьего компонента **Report** установить свойства **DataSet** на необходимые. В свойствах (*настроить бэнды*) включить разделы: ***заголовок отчета, данные***.
2. В разделе **заголовок отчета** разместить метку (компонент **Текст**). В свойствах изменить ее внешний вид и подпись «Отчет по авторам и написанным книгам».
3. Вызвать мастер группировки. Панель *Отчёт -> Мастер группировки*. В качестве условия группировки указать поле, по которому будет осуществляться группировка данных: Autors -> name_autor. Нажать *Добавить*.
4. В результате получим бэнды: Заголовок группы (содержит имя автора), Данные, Подвал группы. На раздел **данные** перетенуть обьекты с панели *данные* по следующему пути: Источники данных -> Autors -> Books -> title_book. Пример представлен на рис. 4.
5. В главной форме приложения в подпункте **Отчет с группировкой по двум таблицам** в методе Click написать команду: **Report3.Show().**

6. Запустить приложение, проверить работу.

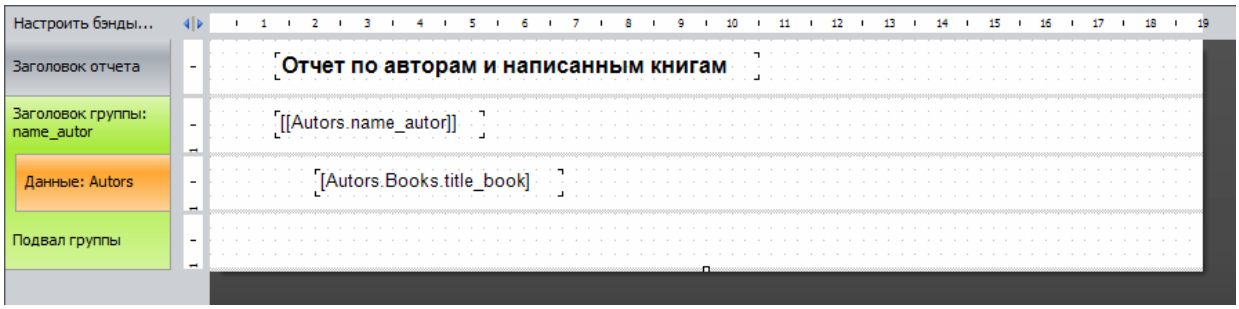


Рис. 4. Пример отчета

Задания к лабораторной работе №9

На Visual C# создать новый проект для индивидуальной БД, созданной в предыдущих лабораторных работах, создать интерфейс, включающий все функции и процедуры, которые описаны по ходу текущей лабораторной работы. Результаты запросов поиска вывести в виде отчетов.